UNISYS

Product Information Announcement

o New Release o Revision • Update o New Mail Code

Title

System 80 OS/3 Data Utilities Operating Guide (System 80) For Series 90 see UP-8069

This Product Information Announcement announces the release and availability of Update B to the *System 80 OS/3 Data Utilities Operating Guide*.

Data utilities, commonly known as the DATA routine, is an OS/3 software component that provides an easy-to-use method for reproducing and maintaining data files. You can run the DATA routine interactively or use it with the card reader in the batch environment.

This guide is intended for the programmer who needs to reproduce and maintain data files. It describes how to use the DATA routine interactively and in the batch environment. Additionally, it describes how to use MILOAD, a special-purpose utility that enhances performance when you load MIRAM files.

This update documents the following additional changes for Release 13:

- Change in the default value of the // PARAM KEYED statement used to produce an empty keyed or an empty sequential MIRAM output file from an empty keyed MIRAM input file.
- Addition of the // EXEC MILOAD statement to the sample job stream shown in Example 5 (Tape Input to Disk Output) of Section 7.

All other changes in this guide are corrections, deletions, or expanded descriptions applicable to items present in the software prior to this release.

You can order the update only, or the complete manual with all updates. To receive only the update, order UP-8834 Rev. 3-B. To receive the complete manual, order UP-8834 Rev. 3.

This document can be ordered through your branch representative or from Unisys Corporation, Corporate Software and Publications Operations, 13250 Haggerty Road North, Plymouth, Michigan 48170.

Announcement only:
MB00, SAB, and SAE

Announcement and attachments:
MBW, MBB1, and MB02

System: System 80
Release: 13
Date: June 1990
Part Number: UP-8834 Rev. 3-B

UNISYS

PUBLICATIONS UPDATE

System 80

OS/3
Data Utilities
Operating Guide

UP-8834 Rev. 3-A

This Library Memo announces the release and availability of Update A to the *OS/3 Data Utilities Operating Guide*, UP-8834 Rev.3.

This guide is a standard library item (SLI). It is part of the standard library provided automatically with the purchase of the product.

Data utilities, commonly known as the DATA routine, is an OS/3 software component that provides an easy-to-use method for reproducing and maintaining data files. You can run the DATA routine interactively or use it with the card reader in the batch environment.

This guide is intended for the programmer who needs to reproduce and maintain data files. It describes how to use the DATA routine interactively and in the batch environment. Additionally, it describes how to use MILOAD, a special-purpose utility that enhances performance when you load MIRAM files.

This update for Release 13.0 documents support for:

- System 80 model 7E
- M9720 disk subsystem

Also, an appendix was added which contains keyword parameters for the DATA routine when it's run in a mixed data management mode.

All other changes in this guide are corrections, deletions, or expanded descriptions applicable to items present in the software prior to this release.

You can order the update only, or the complete manual with the update. To receive only the update, order UP-8834 Rev. 3-A. To receive the complete manual, order UP-8834 Rev. 3.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists MB00, SAB, and SAE	Mailing Lists MB02 and MBW (32 pages plus Memo)	Library Memo for UP-8834 Rev. 3-A
		RELEASE DATE: January 1990

UNISYS

PUBLICATIONS REVISION

System 80

OS/3
Data Utilities
Operating Guide

UP-8834 Rev. 3

This Library Memo announces the release and availability of the *OS/3 Data Utilities Operating Guide*, UP-8834 Rev.3.

This guide is a standard library item (SLI). It is part of the standard library provided automatically with the purchase of the product.

Data utilities, commonly known as the DATA routine, is a software component of Unisys Operating System/3 (OS/3). It provides an easy-to-use method for reproducing and maintaining data files. You can run the DATA routine interactively or use it with the card reader in the batch environment.

This guide is intended for the programmer who needs to reproduce and maintain data files. It describes how to use the DATA routine interactively and in the batch environment. Additionally, it describes how to use MILOAD, a special-purpose utility that enhances performance when you load MIRAM files.

Changes to this guide for Release 12.0 include:

- SPL, a new parameter in the I@DATA job control stream allowing you to specify that spool files produced by interactive data utilities be held for printing at a later time
- A new option in the interactive data utility allowing you to specify that your primary disk file is shareable (ACCESS=SRD)
- MKR, a new keyword parameter for MILOAD allowing you to organize data in any key sequence in the OUTPUT1 file
- REBUILD, a new keyword parameter for MILOAD allowing you to rebuild an existing file's index
- A new method for handling illegal duplicate keys in MILOAD

All other changes in this guide are corrections, deletions, or expanded descriptions applicable to items present in the software prior to this release

Additional copies may be ordered through your Unisys representative.

Destruction Notice: This revision supersedes and replaces *OS/3 Data Utilities User Guide*, UP-8834 Rev. 2 released on Library Memo dated June 1983. Please destroy all copies of UP-8834 Rev. 3, its updates, and Library memos.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists MBZ, MCZ, and MMZ	Mailing Lists MB00, MB02, and MBWA (170 pages plus Memo)	Library Memo for UP-8834 Rev. 3
		RELEASE DATE: October 1988

UNISYS

System 80 OS/3 Data Utilities

Operating Guide

October 1988

Printed in U S America
UP-8834 Rev. 3

Priced Item

UNISYS

System 80 OS/3 Data Utilities

Operating Guide

OS/3 Release 13

June 1990

Printed in U S America
UP-8834 Rev. 3 - Update B

Priced Item

NO WARRANTIES OF ANY NATURE ARE EXTENDED BY THIS DOCUMENT. Any product and related material disclosed herein are only furnished pursuant and subject to the terms and conditions of a duly executed Program Product License or Agreement to purchase or lease equipment. The only warranties made by Unisys, if any, with respect to the products described in this document are set forth in such License or Agreement. Unisys cannot accept financial or other responsibility that may be the result of your use of the information in this document or software material, including direct, indirect, special, or consequential damages.

You should be very careful to ensure that the use of this information and/or software material complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

Correspondence regarding this publication should be forwarded to Unisys Corporation either by using the Business Reply Mail form at the back of this manual or by addressing remarks directly to Unisys Corporation, OS/3 Systems Product Information Development, P.O. Box 500, Mail Station E5-114, Blue Bell, Pennsylvania, 19424, U.S.A.

PAGE STATUS SUMMARY
ISSUE: Update B - UP-8834 Rev. 3

Part/Section	Page Number	Update Level
Cover		A
Title Page/Disclaimer		B
PSS	iii	B
About This Guide	v	Orig.
	vi, vii	A
	viii thru x	Orig.
Contents	xi, xii	Orig.
	xiii	A
	xiv thru xvi	Orig.
	xvii	A
Part 1	Tab Breaker	Orig.
1	1,2	Orig.
	3,4	A
Part 2	Tab Breaker	Orig.
2	1 thru 15	Orig.
3	1 thru 12	Orig.
Part 3	Tab Breaker	Orig.
4	1 thru 6	Orig.
	7	A
	8	B
	9 thru 35	Orig.
	36	A
	37 thru 55	Orig.
5	1 thru 9	Orig.
6	1 thru 22	Orig.

Part/Section	Page Number	Update Level
Part 4	Tab Breaker	
7	1,2	Orig.
	3	A
	4 thru 14	Orig.
	15	B
	16, 17	Orig.
Appendix A	Tab Breaker	A
	1 thru 6	A
Index	Tab Breaker	A
	1	A
	2 thru 8	Orig.
User Comments Form		
Back Cover		

Part/Section	Page Number	Update Level

Technical changes are denoted by a change bar in the margin.

About This Guide

Purpose

This guide instructs the Unisys System 80 programmer in the use of the Unisys Operating System/3 (OS/3) data utilities program.

Scope

This guide:

- Introduces the data utilities program and briefly describes its uses, characteristics, processing methods, and operating environments
- Describes the interactive execution of the data utilities program
- Describes how to use the data utilities program in a batch environment
- Introduces and describes how to use MILOAD, a special-purpose utility that enhances performance when loading large, multikeyed MIRAM files

Audience

The primary audience for this guide is the programmer who needs to reproduce and maintain data files.

Prerequisites

Anyone using this guide should be familiar with the Unisys Operating System/3 (OS/3).

How to Use This Guide

As you read through the guide you'll notice that you can run the data utilities program interactively via a workstation or use it with a card reader in the batch environment. You'll also note that information on how to use the MILOAD utility is included.

The guide is divided into parts that deal specifically with each of these areas.

When you decide what you want to do, read the parts of the guide that apply.

Organization

This guide is divided into four parts, an appendix, and an index.

Part 1. Introduction

This part consists of one section:

- **Section 1. Data Utilities Program**

  This section introduces you to the data utilities program, gives examples of possible applications, describes file organization, and dicusses the proceesing environments.

Part 2. File Processing In An Interactive Environment

This part consists of two sections:

- **Section 2. Interactive Data Utility**

  This section explains how to initiate an interactive dialog and provides information on interactive processing considerations, error messages, output listings, and default selection.

- **Section 3. Sample Interactive Dialog**

  This section provides a typical interactive dialog.

Part 3. File Processing In A Batch Environment

This part consists of three sections:

- **Section 4. Batch Processing Considerations**

  This section describes the job control stream requirements and the data utilities control statements in a batch environment.

- **Section 5. Control Streams**

  This section contains sample job control streams that illustrate how to perform typical data utilities program functions in a batch environment.

- **Section 6. Job Control Procedures (Jprocs)**

  This section describes the jprocs supplied by Unisys that allow you to perform various data utilities program functions and provide examples of their use in a batch environment.

Part 4. Loading MIRAM Files In A Batch Environment

This part consists of one section:

- **Section 7. MILOAD Utility - Special Purpose Loader For MIRAM Files**

  This section describes the MILOAD utility and provides examples of its use.

The appendix contains keyword parameters for the DATA routine when it's run in a mixed data management mode.

Notation Conventions

Here are descriptions of the conventions you must follow to properly code the statements to express the DATA routine operations you desire. Coding conventions for subparameters, optional and required keyword parameters, and punctuation marks are given. The use of capital and lowercase letters in the DATA routine statements is explained.

The conventions used to present the DATA routine statements are as follows:

- Information that must be coded exactly as shown is presented in uppercase letters. This information denotes control statement mnemonics or keyword parameters of a DATA routine utility statement.

  Examples

```
UCD A=(80,80),DC,I1,K1,OS
SEL D=1
COR N=(1),A=(5),B=(6)
```

- Information that must be supplied by the user is presented in lowercase letters.

  Examples

```
Q=(c,s,n,i)
X=(r,s)
```

- Field select (FS) statement parameters are positional and must be coded in the order shown in the description of the FS statement in "Field Select Statement-Moving or Deleting Input Record Fields" in Section 4.

  Examples

```
FS    a,b,c[,d]/CV/a,b,c[,d]
FS    CV/a,b,c[,d]/a,b,c[,d]
FS    a,b,c[,d]/a,b,c[,d]/CV
```

- Information contained within braces represents alternate choices, only one of which may be chosen.

  Examples

  ```
  IN= { vol-ser-no }
      { RES        }
      { RUN        }
  ```

- Information contained within brackets represents optional entries that may be included or omitted. Braces within brackets signify that one of the specified entries must be chosen if that parameter is specified.

  Examples

  ```
  [ , { DC } ]
      { DP }

  [ { FF } ]
    { FV }
  ```

- Commas, virgules (/), and parentheses must be coded exactly as shown.

  Examples

  ```
  UCC A=(r,b),B=(r,b)
  FS a,b,c[,d]/CV/a,b,c[,d]
  ```

- An ellipsis (a series of three periods) indicates the omission of a variable number of entries.

  Example

  ```
  A<(sssss,aa...a)
  ```

- A keyword parameter may contain a sublist of parameters called subparameters, which are separated by commas and are positional within parentheses.

  Examples

  ```
  Q=(c,s,n,i)
  FS a,b,c[,d]/a,b,c[,d]/a,b,c[,d]
  ```

- Commas are required when positional subparameters are omitted, except after the last subparameter specified.

  Examples

  ```
  Q=(c,s,,i)
  Q=(c,s)
  ```

- Subparameters contained within brackets are optional. If included, they must be coded in the order shown. If omitted, a comma is required in their place.

  Example

  ```
  OM=(I[,n][,V][,R])
  ```

  can be coded as

  ```
  OM=(I,,V)
  ```

  or:

  ```
  OM=(I,2)
  ```

- When a default specification occurs in the format description, it is printed on a shaded background. If, by parameter omission, the operating system performs processing other than parameter insertion, it is explained under an "if omitted" heading in the parameter description.

  Examples

  $\left\{\begin{matrix}\text{FF}\\\text{FV}\end{matrix}\right\}$

  $\left\{\begin{matrix}\text{K1}\\\text{K2}\end{matrix}\right\}$

- Workstation screen entries made by the user are shown in reverse print (5).

- Parameters may be coded through column 71 and must be followed by at least one blank column.

- The statement identifier (Uio, FS, SEL, DEL, COR, and PAR) may be coded in any columns between 1 and 71. It must be followed by at least one blank space and, unless starting in column 1, preceded by at least one blank space. The Hn statement must start in column 2.

- With the exception of the Hn statement, statements cannot be continued. Additional parameters are coded by repeating the statement identifier.

  Examples

```
UDD B=(80,132),DP,G=(10),H=(5000)
UDD MY,OI,R=(1000),S2,TD
```

  Hn statements are continued by coding a nonblank character in column 72 and resuming the character string in column 2 of a second card. (See "Title Statement - Printing Page Headings" in Section 4 for coding rules for the Hn statement.)

Related Product Information

Note: *Throughout this guide, when we refer you to another manual, use the version that applies to the software level at your site.*

Interactive Services Operating Guide **(UP-9972)**

This document describes the procedures used to communicate interactively through a local workstation or remote terminal with the operating system.

Contents

PART 3. FILE PROCESSING IN A BATCH ENVIRONMENT

Section 4. Batch Processing Considerations

Figures

Tables

Section 1
Data Utilities Program

The data utilities program, commonly known as the DATA routine, is a straightforward, easy-to-use method for reproducing and maintaining data files. You can run the DATA routine interactively via a workstation or use it with a card reader in the batch environment. Both the batch and interactive methods describe your files to the DATA routine and inform it of the type of processing to be accomplished. We will concentrate on the interactive method because it is more efficient.

During an interactive execution, you need only sit at a workstation and answer the questions (referred to as a dialog) displayed on the screen. After the screens are displayed and questions answered, the DATA routine is executed. There is no need to know job control language, since all your devices are assigned via the dialog.

1.1. Data Routine Uses

The following are a few examples of possible applications of the DATA routine:

- You might want to transfer data to a different type of device for the purpose of long range storage, or to take advantage of a faster access device.
- Data from one program can be used in another program by reformatting the records and transcribing them to a storage medium compatible with the receiving program.
- You may select or delete specific areas of a file for testing purposes or report preparation.
- If, for some reason, you need to know the contents of a file, it is readily available to you through the use of the print option.
- To make certain that no discrepancies have occurred during a copy operation, you can compare the input file to the output file.

1.2. Device Unit Combinations

In the process of copying data from one device to another device of the same type, you can move data from:

- card to card
- tape to tape
- disk to disk
- diskette to diskette

You may also transcribe data from one type of storage medium to another. The combinations of devices you can use are:

Card

- Card to tape
- Card to disk
- Card to printer
- Card to diskette

Disk

- Disk to card
- Disk to tape
- Disk to printer
- Disk to diskette

Tape

- Tape to card
- Tape to tape
- Tape to printer
- Tape to diskette

Diskette

- Diskette to card
- Diskette to tape
- Diskette to printer
- Diskette to disk

Input and output data from the Unisys diskette subsystem are processed in the same manner as the disk. Therefore, any reference to the disk in the DATA routine will usually apply to the diskette subsystem.

1.3. File Organization

Like all other Unisys System 80 software components, the DATA routine supports multiple indexed random access method (MIRAM) files. These files can be indexed or nonindexed.

Indexed file records are accessed via a key in the index partition of your file, where the keys point to the record before the record is processed. An indexed file record can be loaded in any order.

Nonindexed file records are accessed as they appear in a file; that is, they are processed sequentially and there are no keys pointing to the records. A nonindexed file record is accessed via the relative record number.

System access technique (SAT) files, including SAT librarian files and the // USE LIB statement, are not supported.

Note: *Users of System 80 models 7E, 8, 10, 15, and 20 have the option of running in a "mixed" data management mode, that is, using both consolidated data management and basic data management. In this way, Series 90 users migrating to System 80 models 7E through 20 can use both the MIRAM access method and those access methods supported by basic data management (SAM, DAM, ISAM, IRAM). Such users should consult the Appendix for keyword parameters needed for the DATA routine with files using the other access methods. With consolidated data management alone (not a mixed system), the only access method available for output disk and format-label diskettes is MIRAM.*

1.4. Interactive Processing

Interactive processing consists of a question and answer session (dialog) that you conduct with the data routine via a workstation. The answers that you give assign the necessary input/output devices, tell the DATA routine what you want done, and cause the DATA routine to be executed.

Interactive processing achieves the same results as batch processing; however, it is much easier to use.

Now, let's take a look at a typical interactive workstation screen generated by the DATA routine. Immediately, you will be able to see its ease-of-use.

```
OUTPUT SCREEN 1 DUS24

PLEASE ENTER YOUR OUTPUT FILE TYPE
1. PRINTER
2. CARD
3. TAPE
4. DISKETTE
5. DISK
ENTER (5)
```

This screen enables you to select your output file type. The dialog asks you to input your choice of the five listed items in the ENTER item. Assuming that you want a disk output file, simply overwrite the default 1 with 5 and press the transmit (XMIT) key.

Note: *Even though DISKETTE is listed as an option on the interactive screen, diskettes are not available to model 7E users.*

The preceding screen represents only one program instruction for data utilities. However, once you have entered all of the other necessary program instructions (other screens), the DATA routine is immediately executed; that is, you don't have to wait for the system operator to execute it for you.

The workstation screen illustrations in this document show default values in shaded font 1 while entries made by the user are shown in reverse print (white type on black background) (5).

1.5. Batch Processing

Batch processing consists of coding a set of instructions that will tell the DATA routine what your processing requirements are. These instructions, known as the job control stream, consist of job control statements that assign the necessary input/output devices and data utilities statements that describe what you want done.

After you have completed your coding, the job control stream must be transcribed onto cards or a diskette. The card deck or diskette is then submitted to the system operator for subsequent execution.

Section 2
Interactive Data Utility

2.1. Purpose and Application

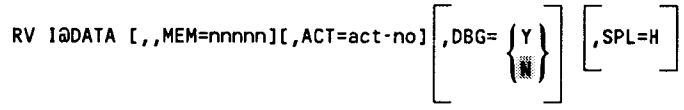
The interactive data utility allows you to interactively execute the DATA routine. This interactive execution consists of a dialog (question and answer session) that you conduct with the DATA routine via a workstation. During the dialog, questions are displayed on the workstation screen and you answer them by typing in the requested information via the workstation keyboard. The answers given define the file processing requirements for the DATA routine.

To use the interactive data utility, you should become familiar with keyins and workstation responses. See the *Screen Format Services Technical Overview* (UP-9977) for workstation operator considerations.

When using the interactive data utility on a remote communications terminal, you must have the field protect feature. If this feature is not used, an SF16 error message is issued.

2.2. Interactive Data Utility Use

To conduct an interactive dialog, you must first log on to the system by entering the LOGON command. After you have successfully logged on, enter the RV I@DATA command via your workstation keyboard. The format of the I@DATA command is:



where:

RV
: Indicates the job is to be run interactively without the use of a card reader.

I@DATA
: Specifies the job name for the interactive data utility.

MEM=nnnnn
: Specifies the minimum main storage requirements (hexadecimal) that are necessary to run your job. The default value is 8000_{16} ($32{,}767_{10}$). This default value is adequate for most processing; however, certain functions that involve tape or disk files may require more. The minimum amount of main storage required can be determined by using the formula described in 2.3.1.

ACT=act-no
: Specifies a 1- to 4-alphanumeric character assigned to you for job accounting purposes.

DBG= { Y / ■ }
: Represents the debug parameter which is the mode of operation used to provide documentation in reporting a user communication form (UCF). It also provides snap dumps of the job region at critical points during execution. Specify Y to run the DATA routine in debug mode. If omitted, N is assumed. Debugging turned on (Y) lengthens interactive response time.

SPL=H
: Specifies that spool files produced by the data utilities program are to be held for printing at a later time. If omitted, the spool files will not be held.

Note: *Prior to using data utilities interactively, you must allocate space for your files by using the ALLOCATE command. For more details, see the* Interactive Services Operating Guide *(UP-9972).*

Once you have successfully initiated the execution, your workstation screen will clear and the first menu selection screen is displayed. This screen will ask you what you want to do. Depending upon your response, either another menu selection screen or a HELP screen (detailed explanation of the menu screen) will appear. Always press the transmit (XMIT) key after entering your processing requests or after requesting HELP.

2.3. Interactive Processing Considerations

To process your files interactively, you may compute the minimum amount of main storage requirements. This main storage requirement is used along with the RUN statement to execute your program. Other important considerations are: error messages, output listings, printer formats, termination information, file statistics, and default selection. These items are discussed in 2.3.1 through 2.3.7.

2.3.1. Minimum Main Storage Requirements

Most executions of the DATA routine can be accomplished in 32K bytes of main storage. However, if you want to compute the exact amount of main storage required to process the data utility routine, use the following formula:

```
M = Maximum of (32768,B)
```

where:

M

Is the minimum amount of main storage required to do this job.

B

Is the total size of the required functional routines, input/output control system (IOCS) modules, and data management buffers. B can be determined by the following formula:

```
B = (19500 + FT + IOT + C + D + E + F)
```

where:

FT

Is the total size of the functional routines required for this job. The sizes of the functional routines are specified in Table 2-1 along with their associated parameters.

IOT

Is the size of all I/O routines for all the files used in this job. Since the printer routine is included in the I/O routines, the printer routine value is not included. The sizes of the I/O routines are specified in Table 2-2.

C

Is the INPUT1 block/buffer size as specified in the second entry of the A=( ) parameter. If the INPUT1 file is a disk file, then the block size is taken from the disk file format labels.

D

Is the OUTPUT1/INPUT2 block/buffer size as specified in the second entry of the B=( ) parameter. If this is a compare operation (K2) and INPUT2 is a disk file, then the block size is taken from the disk file format labels.

E

Is for INPUT1 disk files and is the index buffer size. It is taken from the disk file format labels.

F

Is for OUTPUT1/INPUT2 files only. It represents the disk file index buffer size and it is calculated by multiplying the second entry in the OR=(I,n) or the OM=(I,n) parameter multiplied by 256 decimal. If this is a compare operation (K2), then this value is taken from the disk file format labels.

Table 2-1. Functional Routine Sizes

Function[1]	Invoking Parameter	Size (Decimal)
Correction	COR statement	5200[2]
Select/delete	SEL/DEL statement	810
Field selection	FS statement	2350[3]
Sequence checking	X=( ) parameter	680
Compare	K2 parameter	3850
Print routine	UCP, UTP, UDP, or DP	3540

Notes:

1 The size is not shown for the copy functional routine (K1 parameter) because the copy function is included in the base size of the formula.

2 Add the maximum INPUT1 file record size to this number.

3 Add the maximum OUTPUT1/INPUT2 file record size to this number.

Table 2-2. I/O Routine Sizes

Type	INPUT1	INPUT2	OUTPUT1	OUTPUT2
Card	290	290	260	260
Tape	480	480	350	-
Diskette MIRAM Disk	650	650	570	-
DCON4 Tape	580	-	-	-

Example

Copy an indexed MIRAM file to tape.

```
UDT A = (128, 256), B = (128, 1024), K1

    B = (19500 + FT + IOT + C + D + E + F)

        FT = 0 because the copy function is included in the 19,500 base size of
             the formula.

       IOT = 650 for MIRAM disk INPUT1 I/O routine
            +350 for tape OUTPUT1 I/O routine
            ----
            1000

         C = 256 for INPUT1 block size
         D = 1024 for OUTPUT1 block size
         E = 256 (1 sector)

         F is not used for tape files.

    B = 19500 + 0 + 1000 + 256 + 1024 + 256

    B = 22036

    M = 32768 or B, whichever is greater

    Since B = 22036

    M = 32768 or X'8000' bytes required for the utility routine
```

2.3.2. Error Messages

If error conditions arise during the execution of the data utilities program, error messages are generated. These error messages are displayed on the printer, the system log file, and on the workstation that initiated the dialog.

The errors generated by the DATA routine are grouped into four categories:

1. Informative
2. Warning
3. Serious
4. Fatal

These categories are reflected in Table 2-3 according to bit settings in the UPSI byte at job termination:

Table 2-3. UPSI Byte Settings

Setting	Category
X'00'	Informative
X'20'	Warning
X'40'	Serious
X'80'	Fatal

Refer to the *System Messages Operations Reference Handbook* (UP-8076) for error messages.

2.3.3. Output Listings

The DATA routine provides the following output listings:

- Listing of output data in different formats
- Termination information
- Automatic display of the file statistics for your job's primary file at termination

2.3.4. Printer Formats

The DATA routine can produce printed copy output in two formats: display or list. By your selection of keyword parameters in the Uio statement, you can designate which format you prefer, and if the character mode should be EBCDIC or hexadecimal. Hexadecimal displays require twice as many print positions as the same data displayed in EBCDIC. Figures 2-1 through 2-6 show examples of the display and list formats.

Display Format

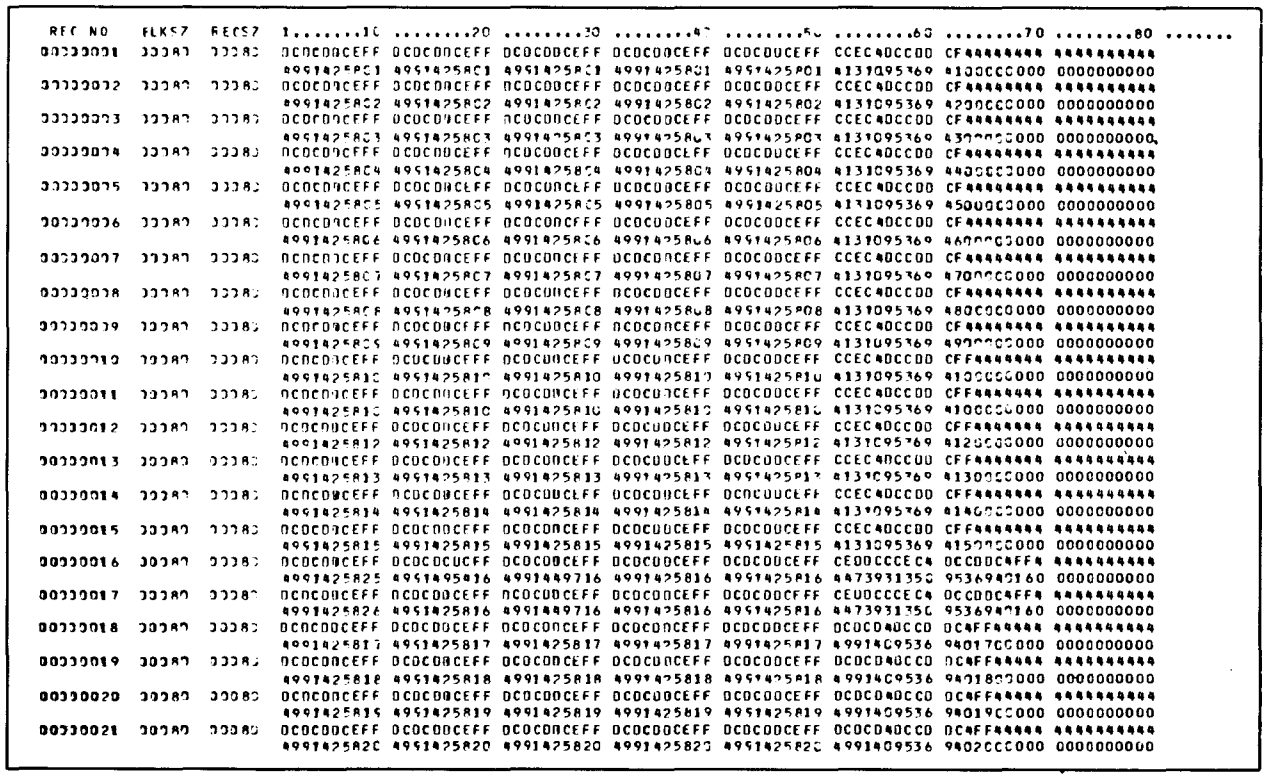
The first 20 print positions of the first line contain column headings. The remainder of the line shows the position of each byte. Subsequent lines show the physical location and size of each block and record, as well as the content of each byte. The columns, record number (REC#), block size (BLKSZ), and record size (RCSZ), represent the following:

Column	Description
REC#	The number of the record relative to the first record of the file. All records are numbered starting with 1.
BLKSZ	The input block size; that is, for fixed-length, nondisk files, this number is taken from the b portion of the *A* keyword parameter (A=(r,b)). For disk files, this number is taken from the VTOC.
RCSZ	For output files, this number is the record length given in the first two bytes of each output record.

Figures 2-1 through 2-3 are examples of the display format in EBCDIC and hexadecimal modes.

```
 REC NO    BLKSZ   RECSZ  1.......10 .......20 .......30 .......40 .......50 .......60 .......70 .......80 .......
00000001  00080  00080  MIRAMKEY01 MIRAMKEY01 MIRAMKEY01 MIRAMKEY01 MIRAMKEY01 DATA RECOR 01
00000002  00080  00080  PIRAMKEY02 PIRAMKEY02 MIRAMKEY02 MIRAMKEY02 MIRAMKEY02 DATA RECOR 02
00000003  00080  00080  MIRAMKEY03 MIRAMKEY03 MIRAMKEY03 MIRAMKEY03 MIRAMKEY03 DATA RECOR 03
00000004  00080  00080  PIRAMKEY04 PIRAMKEY04 MIRAMKEY04 MIRAMKEY04 MIRAMKEY04 DATA RECOR 04
00000005  00080  00080  MIRAMKEY05 MIRAMKEY05 MIRAMKEY05 MIRAMKEY05 MIRAMKEY05 DATA RECOR 05
00000006  00080  00080  PIRAMKEY06 PIRAMKEY06 MIRAMKEY06 MIRAMKEY06 MIRAMKEY06 DATA RECOR 06
00000007  00080  00080  MIRAMKEY07 MIRAMKEY07 MIRAMKEY07 MIRAMKEY07 MIRAMKEY07 DATA RECOR 07
00000008  00080  00080  PIRAMKEY08 PIRAMKEY08 MIRAMKEY08 MIRAMKEY08 MIRAMKEY08 DATA RECOR 08
00000009  00080  00080  MIRAMKEY09 MIRAMKEY09 MIRAMKEY09 MIRAMKEY09 MIRAMKEY09 DATA RECOR 09
0000000A  00080  00080  PIRAMKEY10 PIRAMKEY10 MIRAMKEY10 MIRAMKEY10 MIRAMKEY10 DATA RECOR 010
0000000B  00080  00080  MIRAMKEY10 MIRAMKEY10 MIRAMKEY10 MIRAMKEY10 MIRAMKEY10 DATA RECOR 010
0000000C  00080  00080  PIRAMKEY12 PIRAMKEY12 MIRAMKEY12 MIRAMKEY12 MIRAMKEY12 DATA RECOR 012
0000000D  00080  00080  MIRAMKEY13 MIRAMKEY13 MIRAMKEY13 MIRAMKEY13 MIRAMKEY13 DATA RECOR 013
0000000E  00080  00080  PIRAMKEY14 PIRAMKEY14 MIRAMKEY14 MIRAMKEY14 MIRAMKEY14 DATA RECOR 014
0000000F  00080  00080  MIRAMKEY15 MIRAMKEY15 MIRAMKEY15 MIRAMKEY15 MIRAMKEY15 DATA RECOR 015
00000010  00080  00080  PIRAMKEY25 PIRAMIN016 MIRAMMIX16 MIRAMKEY16 MIRAMKEY16 DUPLICATE  RECORD 16
00000017  00080  00080  MIRAMKEY26 MIRAMKEY16 MIRAMMIX16 MIRAMKEY16 MIRAMKEY16 DUPLICATE  RECORD 16
00000018  00080  00080  PIRAMKEY17 PIRAMKEY17 MIRAMKEY17 MIRAMKEY17 MIRAMKEY17 MIRAM RECO RD 17
00000019  00080  00080  MIRAMKEY18 MIRAMKEY18 MIRAMKEY18 MIRAMKEY18 MIRAMKEY18 MIRAM RECO RD 18
0000001A  00080  00080  PIRAMKEY19 PIRAMKEY19 MIRAMKEY19 MIRAMKEY19 MIRAMKEY19 MIRAM RECO RD 19
0000001B  00080  00080  MIRAMKEY20 MIRAMKEY20 MIRAMKEY20 MIRAMKEY20 MIRAMKEY20 MIRAM RECO RD 20
```

Figure 2-1. EBCDIC Mode Display Format



```
 REC NO    FLKS?  REGS?  1.......10 .......20 .......30 .......40 .......50 .......60 .......70 .......80 .......
00330031   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CF4444444 4444444444
                         4991425RC1 4991425RC1 4991425RC1 4991425RC1 4991425R01 4131095369 413DCCCC000 0000000000
00330032   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CF4444444 4444444444
                         4991425RC2 4991425RC2 4991425RC2 4991425RC2 4991425R02 4131095369 420DCCC000 0000000000
00330033   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CF4444444 4444444444
                         4991425RC3 4991425RC3 4991425RC3 4991425RC3 4991425R03 4131095369 430CCCC000 0000000000
00330034   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CF4444444 4444444444
                         4991425RC4 4991425RC4 4991425RC4 4991425RC4 4991425R04 4131095369 440CCCC000 0000000000
00330035   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CF4444444 4444444444
                         4991425RC5 4991425RC5 4991425RC5 4991425RC5 4991425R05 4131095369 450CCCC000 0000000000
00330036   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CF4444444 4444444444
                         4991425RC6 4991425RC6 4991425RC6 4991425RC6 4991425R06 4131095369 460CCCC000 0000000000
00330037   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CF4444444 4444444444
                         4991425RC7 4991425RC7 4991425RC7 4991425RC7 4991425R07 4131095369 470CCCC000 0000000000
00330038   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CF4444444 4444444444
                         4991425RC8 4991425RC8 4991425RC8 4991425RC8 4991425R08 4131095369 480CCCC000 0000000000
00330039   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CF4444444 4444444444
                         4991425RC9 4991425RC9 4991425RC9 4991425RC9 4991425R09 4131095369 490CCCC000 0000000000
00330040   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CFF444444 4444444444
                         4991425R10 4991425R10 4991425R10 4991425R10 4991425R10 4131095369 410CCCC000 0000000000
00330041   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CFF444444 4444444444
                         4991425R1C 4991425R1C 4991425R1C 4991425R1C 4991425R1C 4131095369 41CCCCC000 0000000000
00330042   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CFF444444 4444444444
                         4991425R12 4991425R12 4991425R12 4991425R12 4991425R12 4131095769 412CCCC000 0000000000
00330043   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CFF444444 4444444444
                         4991425R13 4991425R13 4991425R13 4991425R13 4991425R13 4131095769 413CCCC000 0000000000
00330044   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CFF444444 4444444444
                         4991425R14 4991425R14 4991425R14 4991425R14 4991425R14 4131095769 414CCCC000 0000000000
00330045   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF CCEC40CCDD CFF444444 4444444444
                         4991425R15 4991425R15 4991425R15 4991425R15 4991425R15 4131095369 415CCCC000 0000000000
00330046   33387  33383  DCDCD0CEFF DCDCDCUCFF DCDCD0CEFF DCDCD0DCFF DCDCD0DCFF CEDDCCCEC4 DCCDDC4FF4 4444444444
                         4991425R25 4991425R16 4991449716 4991425816 4991425R16 4473931350 9536940160 0000000000
00330047   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CFFF CEUDCCCEC4 DCCDDC4FF4 4444444444
                         4991425826 4951425816 4991449716 4991425816 4991425R16 4473931350 9536940160 0000000000
00330048   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0DCFF DCDCD4DCCD DC4FF44444 4444444444
                         4991425817 4951425817 4991425817 4991425817 4991425R17 4991409536 9401700000 0000000000
00330049   33387  33383  DCDCD0DCFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD4DCCD DC4FF44444 4444444444
                         4991425818 4951425818 4991425818 4991425818 4991425R18 4991409536 9401800000 0000000000
00330050   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD4DCCD DC4FF44444 4444444444
                         4991425819 4951425819 4991425819 4991425819 4991425R19 4991409536 9401900000 0000000000
00330051   33387  33383  DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD0CEFF DCDCD4DCCD DC4FF44444 4444444444
                         4991425R2C 4951425R20 4991425820 4991425820 4951425R2C 4991409536 9402CCC000 0000000000
```

Figure 2-2. Hexadecimal Mode Display Format

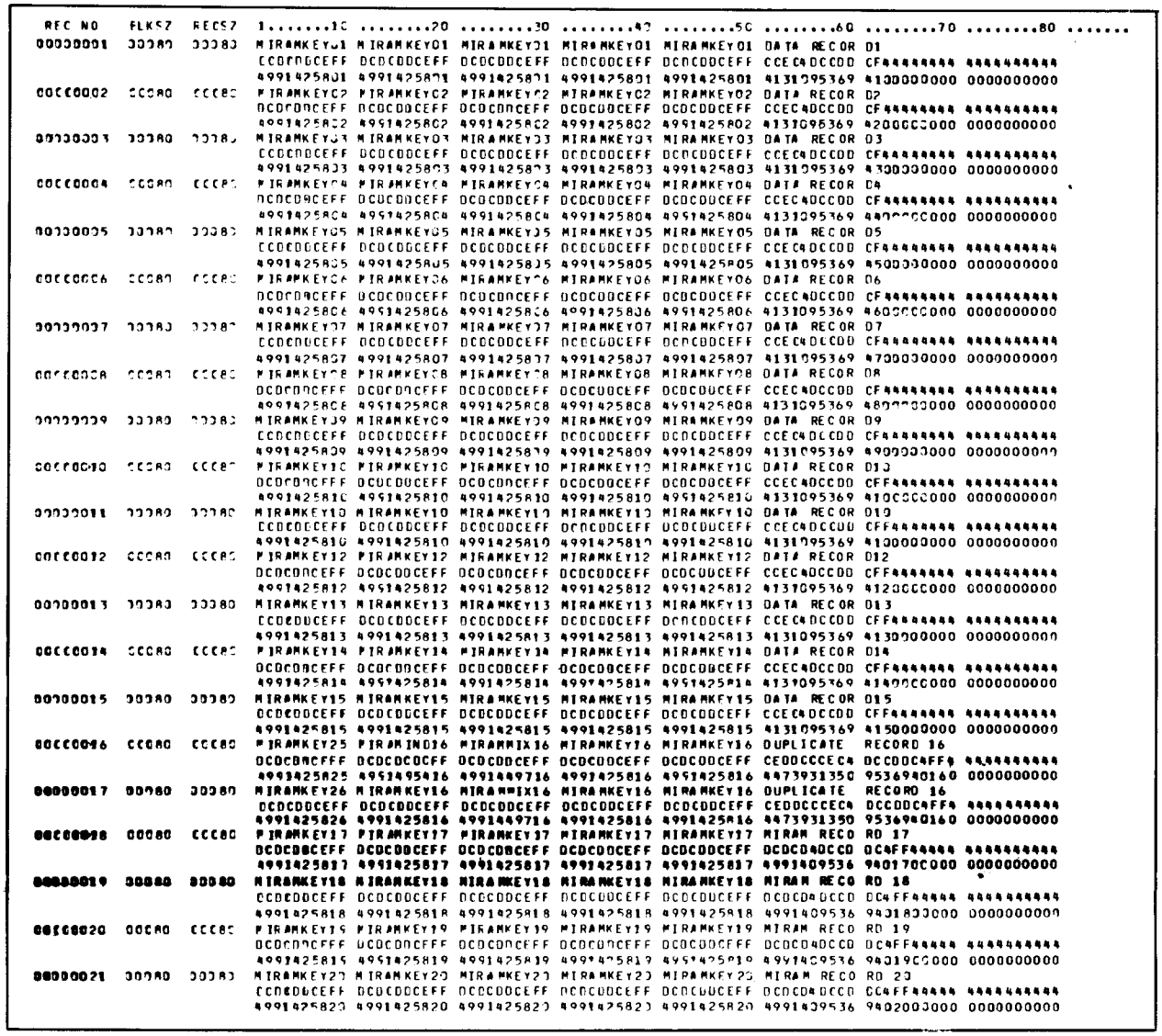


Figure 2-3. Combination of EBCDIC and Hexadecimal Mode Display Format

List Format

Displays your files, byte by byte. When you request the list format in combination EBCDIC and hexadecimal, the position of each byte is printed on the following line. Examples are shown in Figures 2-4 through 2-6.

```
MIRAMKEY01MIRAMKEY01MIRAMKEY01MIRAMKEY01MIRAMKEY01DATA  RECORD1
MIRAMKEY02MIRAMKEYC2MIRAMKEYC2MIRAPKEYC2MIRAMKEYC2DATA  RECORD2
MIRAMKEY03MIRAMKEY03MIRAMKEY03MIRAMKEY03MIRAMKEY03DATA  RECORD3
MIRAMKEY04MIRAMKEYC4MIRAMKEY04MIRAPKEY04MIRAMKEYC4DATA  RECORD4
MIRAMKEY05MIRAMKEY05MIRAMKEY05MIRAMKEY05MIRAMKEY05DATA  RECORD5
MIRAMKEY06MIRAMKEYC6MIRAMKEY06MIRAPKEYC6MIRAMKEY06DATA  RECORD6
MIRAMKEY07MIRAMKEY07MIRAMKEY07MIRAMKEY07MIRAMKEY07DATA  RECORD7
MIRAMKEY08MIRAMKEYC8MIRAMKEY08MIRAPKEY08MIRAMKEYC8DATA  RECORD8
MIRAMKEY09MIRAMKEY09MIRAMKEY09MIRAMKEY09MIRAMKEY09DATA  RECORD9
MIRAMKEY1CMIRAMKEY1CMIRAMKEY10MIRAPKEY1CMIRAMKEY1CDATA  RECORD10
MIRAMKEY1DMIRAMKEY1DMIRAMKEY10MIRAMKEY1DMIRAMKEY1DDATA  RECORD1D
MIRAMKEY12MIRAMKEY12MIRAMKEY12MIRAPKEY12MIRAMKEY12DATA  RECORD12
MIRAMKEY13MIRAMKEY13MIRAMKEY13MIRAMKEY13MIRAMKEY13DATA  RECORD13
MIRAMKEY14MIRAMKEY14MIRAMKEY14MIRAPKEY14MIRAMKEY14DATA  RECORD14
MIRAMKEY15MIRAMKEY15MIRAMKEY15MIRAMKEY15MIRAMKEY15DATA  RECORD15
MIRAMKEY25MIRAMINC16MIRAMPIX16MIRAPKEY16MIRAMKEY16DUPLICATE RECORD 16
MIRAMKEY26MIRAMKEY16MIRAMMIX16MIRAMKEY16MIRAMKEY16DUPLICATE RECORD 16
MIRAPKEY17MIRAMKEY17MIRAMKEY17MIRAPKEY17MIRAMKEY17MIRAM RECORD 17
MIRAMKEY18MIRAMKEY18MIRAMKEY18MIRAMKEY18MIRAMKEY18MIRAM RECORD 18
MIRAPKEY19MIRAMKEY19MIRAMKEY19MIRAPKEY19MIRAMKEY19MIRAM RECORD 19
MIRAMKEY20MIRAMKEY20MIRAMKEY20MIRAMKEY20MIRAMKEY20MIRAM RECORD 20
```

Figure 2-4. EBCDIC Mode List Format

Figure 2-5. Hexadecimal Mode List Format

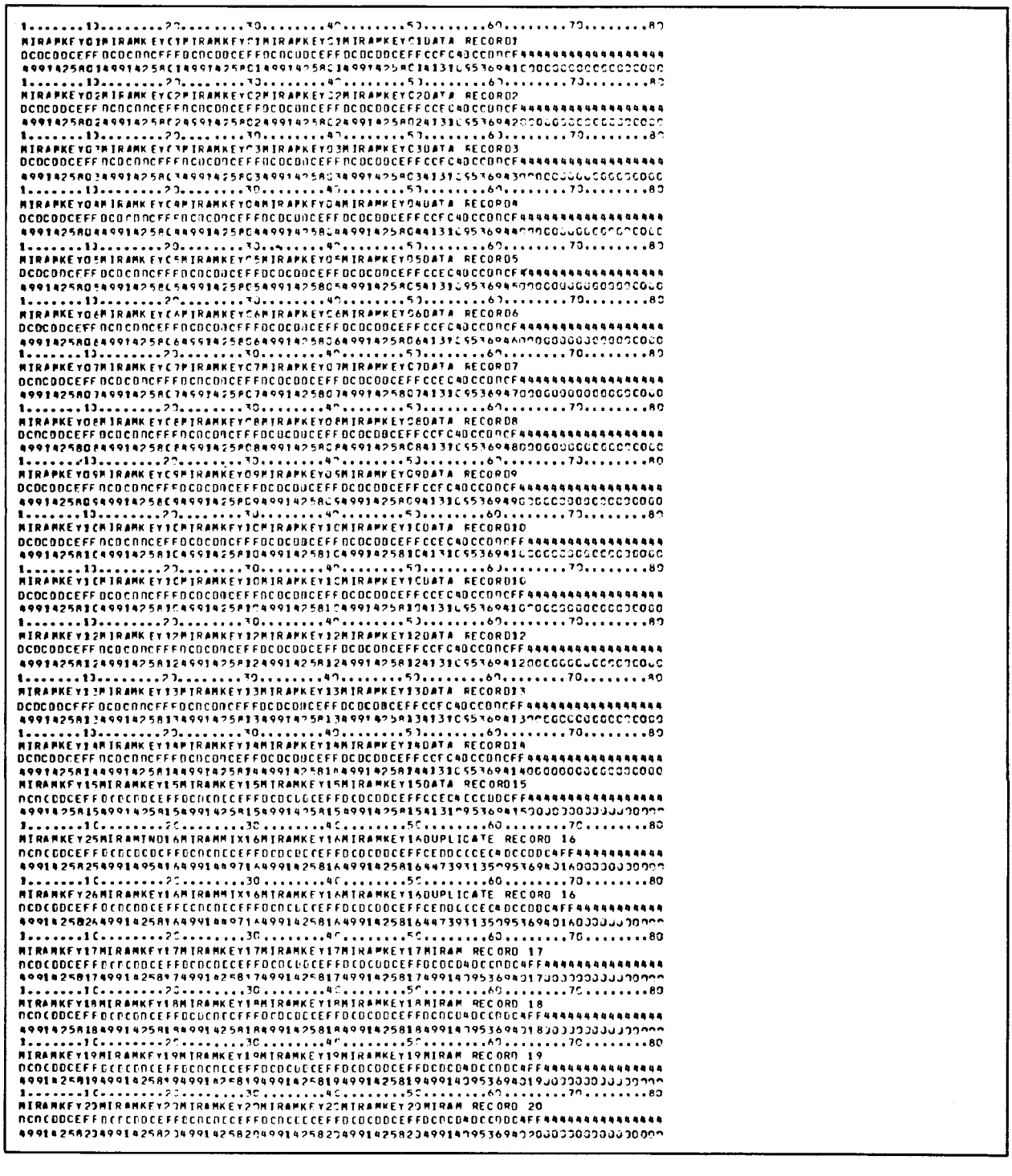


Figure 2-6. Combination of EBCDIC and Hexadecimal Mode List Format

2.3.5. Termination Information

- Card, tape, printer files

  When you execute a program using an interactive execution, the INPUT1/OUTPUT1 format of the termination information for card, tape, and printer files is:

```
INPUT1/OUTPUT1...(FILENAME)..... { CARD    }
                                 { TAPE    }
                                 { PRINTER }
RECORD SIZE.....nnnnn
BLOCK/BUFFER SIZE.....nnnnn
RECORD FORMAT..... { FIXBLK }
                   { FIXUNB }
RECORD FORMAT      { VARBLK }
                   { VARUNB }
FILE ORG....SAM    RCDS UNEQUAL    nnnnnnn
```

  The format for the INPUT2 file is:

```
INPUT2...(FILENAME)..... { CARD    }
                         { TAPE    }
                         { PRINTER }
RECORD SIZE.....nnnnn
BLOCK/BUFFER SIZE.....nnnnn
RECORD FORMAT..... { FIXBLK }
                   { FIXUNB }
RECORD FORMAT      { VARBLK }
                   { VARUNB }
FILE ORG....SAM    RCDS UNEQUAL    nnnnnnn
```

- MIRAM disk and diskette files

  The format of the termination information for MIRAM INPUT1 and OUTPUT1 disk and/or diskette files is:

```
INPUT1/OUTPUT1...(FILENAME)...DISK
RECORD SIZE.....nnnnn
BLOCK/BUFFER SIZE.....nnnnn
MIRAM KEY    LOC   LEN  CHG  DUP
   KEY1      nnnnn nnnnn Y/N  Y/N
   KEY2      nnnnn nnnnn Y/N  Y/N
   KEY3      nnnnn nnnnn Y/N  Y/N
   KEY4      nnnnn nnnnn Y/N  Y/N
   KEY5      nnnnn nnnnn Y/N  Y/N
INDEX BUFFER SIZE.....nnnnn
RECORD CONTROL BYTE.....YES/NO
RECORD SLOT SIZE.....nnnnn
```

```
DISK SECTOR SIZE.............nnnnn
VOLUME MOUNT SETTING.........  SINGLE/MULTI
RECORD FORMAT................{FIXBLK   }
                             {FIXUNBLK }
                             {VARBLK   }
                             {VARUNBLK }
FILE ORG....MIRAM {CONSEC} RCDS UNEQUAL...nnnnnnnn
                  {INDEX }
                  {MIXED }
```

The format for the INPUT2 file is:

```
INPUT2...(FILENAME)...........DISK
RECORD SIZE..................nnnnn
BLOCK/BUFFER SIZE............nnnnn
MIRAM KEY   LOC   LEN   CHG  DUP
   KEY1     nnnnn nnnnn  Y/N  Y/N
   KEY2     nnnnn nnnnn  Y/N  Y/N
   KEY3     nnnnn nnnnn  Y/N  Y/N
   KEY4     nnnnn nnnnn  Y/N  Y/N
   KEY5     nnnnn nnnnn  Y/N  Y/N
INDEX BUFFER SIZE............nnnnn
RECORD CONTROL BYTE..........YES/NO
RECORD SLOT SIZE.............nnnnn
DISK SECTOR SIZE.............nnnnn
VOLUME MOUNT SETTING.........SINGLE/MULTI
RECORD FORMAT................{FIXBLK   }
                             {FIXUNBLK }
                             {VARBLK   }
                             {VARUNBLK }
FILE ORG...MIRAM {CONSEC} RCDS UNEQUAL..nnnnnnnn
                 {INDEX }
                 {MIXED }
```

2.3.6. File Statistics

The following message displays the file statistics for your job at termination.

```
DU99 -I- IN1 record-count {IN2} record-count UPSI=xx
                          {OUT}
```

2.3.7. Default Selection

When you use the DATA routine interactively, the dialog asks you to specify certain values or choices. In many instances, the routine provides defaults for these values and choices. The default values prevail unless you override them. Therefore, if a default value does not meet your needs, override it when the input is requested.

For example, when copying variable MIRAM disk/diskette files, you are asked to enter three sizes for the output file. The first is the minimum record size of the output variable record; the default is the size specified in the input file labels. The second is the output buffer size; the default is the buffer size also specified in the input file labels. The third is the maximum output record size; the default is the value specified for the output buffer size. If you do not overwrite the output buffer value in your second selection, the maximum output record size defaults to the buffer size specified in the input file labels.

When you copy a file and your output is on disk or diskette, the following output file characteristics do not default to those of the input file:

- Index sector size
- Record attribute
- Write protection and verification indicators
- Record control byte
- Single volume mount specifications

You must specify these characteristics during the dialog.

Section 3
Sample Interactive Dialog

3.1. Diskette-to-Disk Copy Operation

A typical diskette-to-disk copy operation is provided in the following description. This program assumes that you have logged on, prepped your disk pack, and allocated file space prior to execution. Also, the program consists of menu selection screens and HELP screens that have already been adjusted to fit your processing needs; that is, all of the SHADED default values have been overwritten with REVERSE type entries. Note that parentheses are protected files and are not entered by the user.

Your diskette input file is in data-set label mode. It has a filename of DUTEST11, has 80-byte unblocked fixed length records, and resides on volume DKST10.

Your disk output file has the filename DUTESTMIRAMKEYED. It has 80-byte, unblocked, fixed-length records and resides on volume D01892. The disk output file will contain three keys and is explained in Step 11.

In addition, duplicate keys are allowed and are marked ALLOWED TO BE CHANGED (in subsequent uses of the file). A record control byte is allocated in each record that also allows records to be logically deleted in subsequent uses of the file. (In this example, the record is not physically deleted). A multivolume file and write verification are used. Finally, the output file is printed.

Once you have chosen your input and output file specifications, you can start the DATA routine using the following command:

```
RV I@DATA ,,MEM=A000,ACT=WXYZ
```

This command calls the DATA routine and assigns 40,960 decimal bytes ($A000_{16}$) of main storage to your job. The DBG and SPL parameters are not specified; therefore, the debug mode is not to be used and the spool files produced are not to be held for printing at a later time.

Step 1. Indicating the Operation

After the run command is processed, the first menu selection screen is displayed:

```
SCREEN 1 DUS01
 DO YOU WISH TO
 1. COPY OR PRINT A FILE
 2. COMPARE TWO FILES
 3. CONVERT OS/4 FILES TO OS/3 DISK FILES
 4. CONVERT A S/32-34 $COPY DISKETTE
 5. HELP
 ENTER (1 THRU 5) (1)
```

Because you want to do a diskette-to-disk copy operation, press the XMIT key because 1 is the default value for copying a file. If you want a compare operation, a conversion operation, or need HELP, enter either a 2, 3, 4, or 5 (overwriting 1), and press the XMIT key.

Step 2. Choosing Primary File Type

After you have indicated the type of operation, this screen will appear:

```
INPUT SCREEN 1 DUS02
 PLEASE ENTER THE TYPE OF YOUR PRIMARY FILE
 1. CARD
 2. TAPE
 3. DISKETTE
 4. DISK
 ENTER (1 THRU 4) (3)
```

Because you are copying from a diskette file, your input (or primary) file is a diskette. Therefore, overwrite the default (1) with a 3 and press the XMIT key.

Step 3. Defining Diskette File Characteristics

After choosing your primary file type, the following screen is displayed:

```
DISK SCREEN 1 DUS63
 PLEASE ENTER YOUR PRIMARY DISK VOLUME SERIAL NUMBER AND
 FILE NAME
 1. VSN(DSKT10) (UNLESS CATALOGED)
 2. FN (DUTEST11.............................)
 3. DO YOU WANT THIS FILE SHAREABLE? (Y OR N) (N)
 4. HELP (ENTER ITEM NUMBER OR 4 FOR ALL) (_)
```

Because you already know your diskette file characteristics, enter the appropriate items here (that is, enter DSKT10 in item 1 and DUTEST11 in item 2). Then, because you don't want this file shareable and don't require any HELP, press the XMIT key.

If you answer item 3 with Y, your primary input file will be shareable (ACCESS=SRD). Data Utilities will be able to read the file if it is being used by another job that has also made the file shareable.

Step 4. Specifying Input File Catalog Password

After you have defined your diskette file characteristics, this screen will appear:

```
CATALOGUE SCREEN 1 DUS54

 1. IF THE INPUT1 FILE IS CATALOGUED AND HAS
    A READ PASSWORD, PLEASE SPECIFY THE
    PASSWORD.
       (......)
 2. NEED HELP (Y OR N)   (_)
```

Because there is no password associated with your file, press the XMIT key.

Step 5. Specifying Multiple Input Volume Serial Numbers

After specifying your catalog password, the following screen is displayed:

```
MULTI-VOLUME SCREEN 1    DUS55
 PLEASE SPECIFY IN ORDER OF PROCESSING THE VSN NAME(S) OF THE
 ADDITIONAL VOLUMES YOU WISH DATA UTILITIES TO PROCESS.
  VSN2:  (......)       VSN3:  (......)     VSN4:  (......)
  VSN5:  (......)       VSN6:  (......)     VSN7:  (......)
  VSN8:  (......)       VSN9:  (......)    VSN10:  (......)
 VSN11:  (......)      VSN12:  (......)    VSN13:  (......)
 VSN14:  (......)      VSN15:  (......)    VSN16:  (......)
 IF YOU NEED HELP PLEASE SPECIFY H FOR HELP. (_)
```

Because your file resides on just one volume, press the XMIT key.

Step 6. Specifying Output File Type

After you have defined the input file characteristics, this screen will appear:

```
OUTPUT SCREEN 1 DUS24
 PLEASE ENTER YOUR OUTPUT FILE TYPE
 1. PRINTER
 2. CARD
 3. TAPE
 4. DISKETTE
 5. DISK
 ENTER (5)
```

Because your requested output device is disk, enter a 5 and press the XMIT key.

Step 7. Selecting Optional Output File Characteristics

After specifying the output file type, the following screen is displayed:

```
OUTPUT SCREEN 2 DUS50

 ENTER THE EXTENSION OR INITIALIZATION
 OPTION (APPLIES TO TAPE, DISKETTE, OR
 DISK OUTPUT ONLY):

 1. FILE TO BE INITIALIZED
 2. FILE TO BE EXTENDED

 ENTER (1 OR 2) (1)
```

Because your program copies to a new disk file from a diskette file, you must initialize the disk file. So, press the XMIT key (assuming the default of 1).

Step 8. Indicating Disk File Characteristics

After you have selected the optional output file characteristics, this screen will appear:

```
DISK SCREEN 1 DUS11
 PLEASE ENTER YOUR DISK VOLUME SERIAL NUMBER AND
 FILE NAME
 1. VSN (______)
 2. FN (_____________________________________________)
 3. HELP (ENTER ITEM NUMBER OR 3 FOR ALL) (_)
```

Before you specify the disk file characteristics, let's suppose you are not certain about item number 1 (the volume serial number) and require assistance. So, you request the appropriate HELP screen (DISK HELP SCREEN DUH1111) by entering 1 in the HELP item and pressing the XMIT key. The HELP screen will read as follows:

```
                * * * DISK HELP SCREEN DUH1111* * *
   ''VSN'' SPECIFIES THE VOLUME SERIAL NUMBER (DISK VOLUME LABEL)
   FOR THE DISK VOLUME. THE VOLUME SERIAL NUMBER
   UNIQUELY IDENTIFIES THE DISK PACK TO THE OPERATING
   SYSTEM. IT IS WRITTEN INTERNALLY (ON THE DISK SURFACE)
   AND POSSIBLY EXTERNALLY (GENERALLY ON A GUMMED LABEL).
   THE VSN CANNOT BE MORE THAN SIX ALPHANUMERIC CHARACTERS
   AND THE FIRST CHARACTER MUST BE ALPHABETIC.
   IF THERE ARE LESS THAN SIX, TRAILING BLANKS WILL BE
   ADDED ON THE RIGHT.

NEED MORE (IF ANY) HELP SCREENS? (Y/N=CONTINUE NORMAL PROCESSING)(N)
```

After you have reviewed the HELP screen and are ready to continue normal processing, type an N for no more HELP and press the XMIT key. The menu screen (DISK SCREEN 1) that you were previously working on will reappear. Make your entries as follows:

```
DISK SCREEN 1  DUS11
 PLEASE ENTER YOUR DISK VOLUME SERIAL NUMBER AND
 FILE NAME
 1. VSN(D01892)
 2. FN (DUTESTMIRAMKEYED.....................)
 3. HELP (ENTER ITEM NUMBER OR 3 FOR ALL) (_)
```

Now that you have entered the specified disk file characteristics, that is, D01892 in item 1 and DUTESTMIRAMKEYED in item 2, press the XMIT key.

Step 9. Specifying Output File Catalog Password

After indicating your disk file characteristics, the following screen is displayed:

```
CATALOGUE SCREEN 1 DUS54

 1. IF THE OUTPUT1 FILE IS CATALOGUED AND HAS
    A WRITE PASSWORD, PLEASE SPECIFY THE
    PASSWORD.
       (......)
 2. NEED HELP (Y OR N)    (_)
```

Because there is no password associated with your file, press the XMIT key.

Step 10. Choosing Disk File Type

After specifying your output file catalog password, the following screen is displayed:

```
DISK SCREEN 3  DUS13
 PLEASE ENTER THE TYPE OF DISK FILE YOU WISH
 TO CREATE
 1. KEYED
 2. UNKEYED
 3. HELP
 ENTER (3)
```

Assuming that you don't understand the difference between keyed and unkeyed files, and need HELP, enter 3 and press the XMIT key. The HELP screen (DISK HELP SCREEN DUH1311) is displayed:

```
            * * * DISK HELP SCREEN DUH1311 * * *
DISK FILES CAN BE CREATED WITH OR WITHOUT KEYS. ''KEYS'' ARE SELECTED
FIELDS WITHIN YOUR RECORD WHICH WILL BE USED TO DISTINGUISH A
PARTICULAR RECORD FROM ANOTHER RECORD. DISK FILES CAN BE PROCESSED
SEQUENTIALLY OR RANDOMLY BY RECORD NUMBER. KEYED FILES CONTAIN AN
INDEX AND CAN BE PROCESSED SEQUENTIALLY OR RANDOMLY. UNKEYED FILES
CAN BE PROCESSED SEQUENTIALLY BY RELATIVE RECORD. THE FILE KEY
CHARACTERISTICS FOR THE OUTPUT FILE DEFAULT TO THE CHARACTERISTICS
IN THE PRIMARY FILE.

NEED MORE (IF ANY) HELP SCREENS? (Y/N=CONTINUE NORMAL PROCESSING) (N)
```

After reviewing the HELP screen and deciding to continue normal processing, type an N and press the XMIT key. The next screen displayed will be the previous menu selection screen (DISK SCREEN 3).

```
DISK SCREEN 3  DUS13
 PLEASE ENTER THE TYPE OF DISK FILE YOU WISH
 TO CREATE
 1. KEYED
 2. UNKEYED
 3. HELP
 ENTER (2)
```

Because your disk file has three keys, enter a 1 and press the XMIT key.

Step 11. Defining Disk Key Characteristics

After you have chosen the disk file type, this screen appears:

```
DISK SCREEN 4  DUS14
 PLEASE ENTER THE KEY DESCRIPTION(S) FOR YOUR FILE
 KEY  LENGTH  LOCATION  DUPLICATES  CHANGES

 NUM                          (Y OR N)   (Y OR N)
 1     (10)    (00000)         (Y)        (Y)
 2     (10)    (00030)         (Y)        (Y)
 3     (05)    (00050)         (Y)        (Y)
 4     (00)    (_____)         (N)        (N)
 5     (__)    (_____)         (N)        (N)
HELP (ENTER LEN, LOC, DUP, CHG, OR ALL) (___)
THE FIRST ZERO KEY LENGTH INDICATES NO MORE KEYS.
```

Because the disk file has three keys with various lengths and locations, enter each appropriately. Also, be certain to indicate whether or not you want duplicates and changes. For key number 1 enter a length of 10, assume the default for location, and enter a Y for duplicates and changes. For key number 2 enter a length of 10, enter a location of 00030, and enter a Y for duplicates and changes. For key number 3 enter a length of 05, enter a location of 00050, and enter Y for duplicates and changes.

The key location is relative to byte 0 of the record; therefore, you can think of the key location as the number of bytes preceding the key.

Because your disk output file only has three keys, enter 00 in LENGTH column of the fourth line to stop the processing of this screen, and then press the XMIT key.

Step 12. Specifying Number of 256-Byte Sectors

After defining the disk key characteristics, the following screen will be displayed:

```
DISK SCREEN 5  DUS15
 PLEASE ENTER THE NUMBER OF 256 BYTE SECTORS YOU
 WISH TO USE FOR EACH INDEX BLOCK
 ENTER (NUMBER OR 0 FOR HELP) (01)
```

This value is arbitrary but must be at least 1, so assume the default, move the cursor over to the rightmost response character (in this case, 1), and press the XMIT key.

Notes:

1. *The default values for this parameter are not the same as for the DATA routine in batch mode. In batch mode, this parameter defaults to the values of the input file; in interactive mode, the default values are as shown.*

2. *The more main storage assigned to this buffer, the faster the file access is performed.*

Step 13. Selecting Optional Record, Volume, and Write Specifications

After you have specified the number of 256-byte sectors needed, this screen appears:

```
DISK SCREEN 6  DUS16
 PLEASE ENTER YOUR DISK RECORD CONTROL BYTE, VOLUME
 MOUNT, AND WRITE VERIFICATION SPECIFICATIONS
 1. RECORD CONTROL BYTE (Y OR N) (Y)
 2. SINGLE VOLUME MOUNT (Y OR N)      (N)
 3. WRITE VERIFICATION CHECK (Y OR N) (N)
 4. HELP (ENTER ITEM NUMBER OR 4 FOR ALL) (_)
```

Just as a reminder, review the following:

- The record control byte allows you to employ the delete facility during normal processing. If you allow deleted records on your file, then you will require a control byte.

- Your output file will be multivolume mount, so the file can be randomly accessed. A single-volume mount denotes that your files have only one volume online at any time. If a file is created with this option, it must be processed likewise.

- The write verification check is a parity (hardware) check of output records after they are written to disk. It increases execution time but ensures that the records are correctly written.

Press the XMIT key because you are assuming all the default values.

Note: The default values for this parameter are not the same as for the DATA routine in batch mode. In batch mode, this parameter defaults to the values of the input file; in interactive mode, the default values are as shown.

Step 14. Indicating Output Record and Block Sizes

After selecting the record, volume, and write specifications, the following screen is displayed:

```
DISK SCREEN 8  DUS52
ENTER YOUR OUTPUT RECORD SIZE AND
BLOCK/BUFFER SIZE
 1. RECORD SIZE (00080)
 2. BLOCK/BUFFER SIZE (00256)
 3. HELP (ENTER ITEM NUMBER OR 3 FOR ALL) (_)
```

Because the input file record size and block size each is 80, assume the default of 00080 for item 1. Also assume the default value for item 2 and press the XMIT key. Note, however, that a buffer size less than 512 is illegal for a MIRAM disk file with a record size of 80. If the illegal default buffer size is transmitted, the DATA routine automatically uses the minimum allowed for the record size specified without generating any errors (such as in this example). Conversely, if a value greater than the allowed minimum is specified, it also is subject to these rules and will be validated by data management.

Step 15. Specifying Multiple Output Volume Serial Numbers

After you have indicated the output record and block sizes, this screen will appear:

```
MULTI-VOLUME SCREEN 1    DUS55
 PLEASE SPECIFY IN ORDER OF PROCESSING THE VSN NAME(S) OF THE
 ADDITIONAL VOLUMES YOU WISH DATA UTILITIES TO PROCESS.
  VSN2:  (______)       VSN3:  (______)     VSN4:  (______)
  VSN5:  (______)       VSN6:  (______)     VSN7:  (______)
  VSN8:  (______)       VSN9:  (______)    VSN10:  (______)
 VSN11:  (______)      VSN12:  (______)    VSN13:  (______)
 VSN14:  (______)      VSN15:  (______)    VSN16:  (______)
 IF YOU NEED HELP PLEASE SPECIFY H FOR HELP. (_)
```

Because your output file is on only one volume, press the XMIT key.

Step 16. Choosing Processing Options

After you have specified the multiple output volume serial numbers, this screen appears:

```
OPTION SCREEN 1  DUS25
 PLEASE SELECT (Y OR N) ANY OPTIONS YOU REQUIRE
 1. FILE REPOSITIONING                    (N)
 2. HALT ON RECORD COUNT                  (N)
 3. CORRECTION BY RECORD NUMBER           (N)
 4. SELECTION/DELETION BY FIELD VALUE     (N)
 5. SEQUENCE CHECKING                     (N)
 6. REARRANGE/CONVERT FIELDS              (N)
 7. SEQUENCE NUMBERING                    (N)
 8. HELP (ENTER ITEM NUMBER OR 8 FOR ALL) (_)
```

Because you don't require any additional processing, assume the default of N for all items and press the XMIT key.

Step 17. Defining Punched Card File

After choosing the processing options, the following screen will be displayed:

```
DUAL CARD SCREEN 1  DUS45
  DO YOU WISH DATA TO PUNCH A CARD FILE CONTAINING THE
  FIRST 80 BYTES OF EACH SECONDARY FILE RECORD?
  ENTER (Y OR N) (N)
```

Because you are not producing a punched card version of the diskette file along with the disk copy, assume the default of N and press the XMIT key.

Step 18. Specifying to Print Secondary File as it is Created

After indicating you don't need a punched card file, this screen appears:

```
DUAL PRINT SCREEN 1  DUS46

DO YOU WISH DATA TO PRINT YOUR SECONDARY FILE
AS IT IS CREATED?

ENTER (Y OR N) (Y)
```

Because we want a printed copy as well as a disk copy of the file, we overwrite the default with a Y and press the XMIT key.

Step 19. Choosing Your Printer Format Option

After specifying whether or not to print the secondary file, this screen appears:

```
PRINTER SCREEN 3  DUS20

 PLEASE ENTER YOUR PRINTER FORMAT OPTION

 1. LIST FORMAT

 2. DISPLAY FORMAT

 3. HELP

 ENTER (1)
```

Because we want the printed copy to be in list format, we assume the default 1 and press the XMIT key.

Step 20. Choosing Your Printer Mode Option

After choosing your printer format option, this screen appears:

```
PRINTER SCREEN 4  DUS21

 PLEASE ENTER YOUR PRINTER MODE OPTION

 1. CHARACTER MODE

 2. HEXADECIMAL MODE

 3. BOTH

 4. HELP

 ENTER (1)
```

Because you want the printed copy to be printed in CHARACTER (EBCDIC) mode, assume the default and press the XMIT key.

Step 21. Concluding with Termination Message

After specifying whether or not to print the secondary file, this final screen is displayed:

```
EOJ SCREEN 1  DUS49
 CONVERSATIONAL PHASE OF DATA UTILITIES COMPLETED
   . . . . EXECUTION STARTED . . . .
```

This is the last screen presented by the DATA routine. When the diskette-to-disk copy operation is completed, data utilities will terminate. Any messages generated will be displayed on the workstation.

Section 4
Batch Processing Considerations

4.1. Purpose and Application

When executing the DATA routine in a batch environment, the same processing considerations used in an interactive environment are used: minimum main storage, error messages, termination information, output listings, and printer formats. (Refer to 2.3 for details.) However, you must create a job control stream that provides the information necessary to execute the DATA routine. You can have your job control stream define your primary (input) file in a device assignment set or as embedded card data.

4.2. Job Control Stream Requirements

A job control stream is necessary to execute the DATA routine in the batch environment. A job control stream consists of:

- // JOB statement
- Device assignment set (// DVC, // VOL, // EXT, // LBL, // LFD)
- // EXEC DATA statement
- Start-of-data statement (/$)
- Data utility control statements
- End-of-data statement (/*)
- End-of-job statement (/&)
- // FIN statement
- PARAM statements

The statements are described in 4.2.1 through "// PARAM KEYED" in this section. In addition, use of a job control procedure (jproc) is discussed in 4.2.11 and a sample job control stream is shown in 4.2.12.

4.2.1. // JOB Statement

The // JOB statement is the first statement in the job control stream. It names the job and specifies the amount of main storage that is required for the job. (See 2.3.1 for main storage requirements.)

4.2.2. Device Assignment Sets

The device assignment sets you use in your job control stream consist of the following statements that you must use in this order:

- DVC

  The DVC job control statement designates the device you require.

- VOL

  The VOL job control statement identifies a tape or disk volume by serial number.

- EXT

  The EXT job control statement is needed only when you allocate disk or diskette space.

- LBL

  The LBL statement identifies a tape or disk file by file identifier.

- LFD

  The LFD statement specifies the logical file name and links your file description with the corresponding data management file definition.

A device assignment set is not required for the card reader if your primary (input) file is embedded card data.

To describe your input/output files to the DATA routine, use the following logical file names as they apply to the devices you assign to your file:

- // LFD PRNTR

  Needed to allocate a printer to the job. If a printer is allocated, the DATA routine prints out the start-of-data (/$) statement, the Uio statement, any utility modifier statements, and the end-of-data (/*) statement as each is read from the control stream. Error messages, if applicable, also are printed.

If a printer is not allocated to the job, no headings or control stream listings are generated. If an error should occur, a data management DMxx error message and a data utilities DU fatal error message are displayed on the system console. If the functions specified require a printer and none is allocated, a fatal error condition terminates the job.

A descriptive listing of all data management and data utility error messages is contained in the current version of the *System Messages Operations Reference Handbook* (UP-8076).

- // LFD INPUT1

  Needed for all runs except embedded card data. Defines the input file for a copy generation or the primary input file for a compare operation. If not present, DATA assumes INPUT1 is embedded card data.

- // LFD INPUT2

  Needed only when a compare operation is specified; defines the secondary input.

- // LFD OUTPUT1

  Needed only for a copy operation when the printer is not the primary output.

- // LFD OUTPUT2

  Needed only for a copy operation when the dual output option (DC) specifies card output; defines the card punch.

The DATA routine also accepts file characteristics from a // DD job control statement. When this // DD statement is present within the device assignment set for a file, the DATA routine uses the information specified in the // DD statement rather than the information provided in the data utilities statements; that is, the // DD statement overrides the data utilities statements. For disk input files, the file characteristics specified in the format labels override the information in both the data utilities statements and the // DD statement.

The // EXEC DATA statement follows the device assignment set and initiates execution of the DATA routine. The parameter for this statement must be DATA.

4.2.3. // EXEC DATA Statement

The // EXEC DATA statement follows the device assignment set and starts the execution of the DATA routine.

4.2.4. Start-of-Data Statement (/$)

This statement follows the // EXEC DATA statement and indicates the start of the data utility control statements.

4.2.5. Data Utilities Control Statements

These statements (the Uio statement and the modifier statements) specify the file processing you want the DATA routine to perform. A description of the data utility control statements is found in 4.3.

4.2.6. End-of-Data Statement (/*)

This statement follows the last data utilities control statement and indicates the end of the control statements.

4.2.7. Embedded Card Data

The INPUT1 file for data utilities can be entered in your control stream as embedded card data. This method eliminates the need for assigning your input files in your job control stream. The INPUT1 file and the Uio statement are inserted between two sets of data delimiters (/$ and /*).

4.2.8. End-of-Job Statement (/&)

This statement indicates the end of the job.

4.2.9. // FIN Statement (Ending the Card Reader Operation)

This statement indicates that no more statements are to be read for this job control stream, and it turns off the card reader.

4.2.10. // PARAM Control Statements

There are six // PARAM control statements that you can supply to data utilities to control operation. They are:

// PARAM CONTROL	// PARAM MODE
// PARAM DISPLAY	// PARAM EOJ
// PARAM DTF	// PARAM KEYED

When supplied, these six six statements must appear immediately following the // EXEC DATA statement, but precede the /$ statement.

// PARAM CONTROL

Use this statement to print data utilities control statements on the final output.

Format

```
// PARAM CONTROL= { YES }
                  { NO  }
```

Parameters

YES
: Specifies that data utilities control statements are printed on the final output.

NO
: Specifies that data utilities control statements are not printed on the final output.

// PARAM MODE

This statement is used to specify the debug mode or the OS/4-to-OS/3 data conversion mode.

- Debug Mode

  Use this statement to specify the debug mode. This mode provides snap dumps of the job region at critical points during execution and is made operative by including the following PARAM statement in the control stream immediately following the // EXEC DATA statement:

  ```
  // PARAM MODE=DBG
  ```

  This mode of operation should only be used to provide documentation in reporting a User Communications Form (UCF).

- OS/4-to-OS/3 Data Conversion Mode

  This mode is used to convert OS/4 disk files to OS/3 disk files. Before using this mode, you must use the OS/4 disk data conversion utility (DCON4) to dump the OS/4 files and their characteristics onto a tape file. Then you use the DATA routine to read the tape and create OS/3 disk files (see the UTD statement, "Tape Input Formats" in this section). To do this, include the following PARAM statement in your control stream immediately following the // EXEC DATA statement:

```
// PARAM MODE=OS4
```

You can also convert OS/4 disk files to OS/3 disk files interactively; DCON4 is used to create the input file.

For a detailed description on the use of DCON4 and the conversion mode, see the *OS/4 to OS/3 Disk Data Conversion Utility User Guide/Programmer Reference,* UP-8606.

// PARAM DISPLAY

Use this statement to specify the files where the termination information (see 2.3.3) is written.

Format

```
// PARAM DISPLAY={P    }
                 {L    }
                 {C    }
                 {NONE }
```

Parameters

P
: Indicates the termination information is listed on the printer file.

L
: Indicates the termination information is written to the system log file.

C
: Indicates that the termination information is displayed on the system console.

NONE
: Indicates no termination information is placed in any of the supported files.

Note: *Any combination (PLC) can be used and commas are not permitted.*

// PARAM EOJ

Use this statement to indicate what occurs when the DATA routine terminates.

Format

```
// PARAM EOJ={CANCEL}
             {UPSI  }
```

Parameters

CANCEL
: Indicates that the DATA routine will terminate with CANCEL if a fatal or serious error is encountered. The UPSI byte is not modified regardless of how the job step terminates.

UPSI
: Indicates that the DATA routine is to terminate normally in all cases, but the UPSI byte is set to X'20' for warning errors, X'40', for serious errors, or X'80', for fatal errors.

// PARAM DTF

Use this statement if you are a System 80 Model 8-20 user processing your DATA routine input or output files with basic data management (DTF) interfaces. Using this statement may speed up the processing of unit record files with large numbers of records.

Note: *This statement can be used only by System 80 Model 8-20 users.*

Specify // PARAM DTF if a DTF interface is more efficient for your specific DATA execution. This PARAM statement does not change the physical characteristics of your files.

Diskette files used with this statement must be basic data exchange (BDE) format, data-set-label mode diskettes. Sector size must be 128 bytes or less, unblocked, unspanned records, no record control byte, and record size of 128 bytes or less.

Format

```
// PARAM DTF={ INPUT1 }
              { INPUT2 }
              { OUTPUT }
              { BOTH   }
```

Parameters

INPUT1
: Indicates that the file you specified as the INPUT1 (4.2.2) file uses the DTF interface.

INPUT2
: Indicates that the file you selected as the INPUT2 file (4.2.2) uses the DTF interface.

OUTPUT
: Indicates that the file you selected as the OUTPUT1 file uses the DTF interface.

BOTH

Indicates:

- If you are performing a compare operation, both the INPUT1 and INPUT2 files use the DTF interface.
- If you are doing a copy operation, the INPUT1 and OUTPUT1 files use the DTF interface.

This parameter does not affect the printer interface, which, in a mixed-mode environment, always defaults to consolidated data management.

Diskettes used for input or output must be in BDE mode. If they are not, the DATA routine defaults to consolidated data management interfaces regardless of what you specify with the // PARAM DTF= parameter.

// PARAM KEYED

Use this statement when you want to produce an empty keyed or an empty sequential MIRAM output file from an empty keyed MIRAM input file.

Format

```
// PARAM KEYED= { YES }
               { NO  }
```

Parameters

YES

Specifies that the key specifications from the MIRAM input file are to be copied to the output file. The output file produced is an empty keyed MIRAM file. This is the default specification.

NO

Specifies that the key specifications from the MIRAM input file are not to be copied to the output file. The output file produced is an empty sequential MIRAM file.

4.2.11. Job Control Procedure Use

As previously mentioned, you must provide a job control stream every time you use the DATA routine. As time goes on, you will notice that you are coding the same sequence of job control statements over and over. This repetitious coding can be avoided by using a single statement (a job control procedure call statement) in your job control stream. This jproc replaces the job control statements that you normally would code. The jproc call statement will generate the proper job control statement sequence for you. Additional details are provided in Section 6.

4.2.12. Sample Job Control Stream

Figure 4-1 is an example of a DATA routine control stream, showing the device assignment set, the EXEC statement, the Uio statement, and the placement of data cards.

```
1.  // JOB TESTFIL,,A000
2.  // DVC 20      // LFD PRNTR
3.  // DVC 30      // LFD INPUT1
4.  // DVC 51      // VOL DSP012
5.  // EXT MI,C,,CYL,8
6.  // LBL WRKFIL3   // LFD OUTPUT1,,INIT
7.  // EXEC DATA
8.  /$
9.  UCD keyword parameters
10. /*
11. /&
12. // FIN
13.      data cards
14. /*
```

Figure 4-1. Sample Job Control Stream for Card-to-Disk Operation

Line 1 shows the job beginning, job name TESTFIL, and the main storage requirements. Line 2 assigns the printer to the job. Line 3 assigns the card reader as the input file with the required file-name of INPUT1. Line 4 assigns the disk-file with the volume serial number DSPO12 to this job. Line 5 obtains disk space by allocating contiguous cylinder space on eight cylinders. Line 6 identifies the output file by file label and the required file name of OUTPUT1.

Line 7 requests the loading and execution of the DATA routine. Line 8 designates the start of control cards. Line 9 is the Uio statement that identifies this job as a card-to-disk operation and would include any necessary keyword parameters.

Line 10 designates the end of control cards. Line 11 indicates the end of job. Line 12 terminates the card reader. Line 13 shows the location of your data cards in the control stream. If your input file is to be spooled, precede your data cards with a // DATA control statement and follow your data cards with a // FIN statement to terminate the card reader. Line 14 designates the end of data.

If you require more detailed information concerning job control statements and their parameters, refer to the *Job Control Programming Guide* (UP-9986).

Figure 4-2 is an example of a DATA routine control stream using embedded card data.

```
1.  // JOB TESTFIL,,A000
2.  // DVC 20        // LFD PRNTR
3.  // DVC 51        // VOL DSP012
4.  // EXT MI,C,,CYL,8
5.  // LBL WRKFIL3   // LFD OUTPUT1,,INIT
6.  // EXEC DATA
7.  /$
8.  UCD keyword parameters
9.  /*
10. /$
11. .
12. .
13. .
14. embedded card data
15. .
16. .
17. .
18. /*
19. /&
20. // FIN
```

Figure 4-2. Sample DATA Routine Control Stream Using Embedded Card Data

Notice that with embedded card data, the device assignment set for the card reader is not needed. Line 10 designates the start of data. Lines 11 through 17 show the location of your embedded card data. Line 18 signifies the end of data. Line 19 indicates the end of job. Line 20 terminates the card reader.

If you require more detailed information concerning embedded card data, refer to the current version of the *Job Control Programming Guide* (Up-9986).

4.3. Data Utilities Control Statements

The format descriptions of the data utilities control statements needed to accomplish all of the DATA routine functions are described in 4.3.1 through "Correction Statement-Correction Records" in this section.

4.3.1. Basic Utility Input and Output Statements

You use a utility input and output statement (Uio) to specify the copy or compare functions that you wish the DATA routine to perform. The Uio statement identifies the input and output devices you require and describes the data format of your files to the DATA routine through the keyword parameters you specify. Keyword parameters allow you to describe:

- The format of your input and output files
- Whether this is a copy or compare function
- The file locations where processing begins
- The number of unequal records accepted, or the number of blocks or records processed before DATA routine termination
- Whether the output record is written to more than one device
- Whether input and output tapes are rewound
- Whether printer mismatches are ignored to terminate the DATA routine

To do this, you must include the appropriate Uio statement in your control stream between the /$ and /* job control statements following the // EXEC DATA job control statement.

Input and Output Statement Mnemonics

The utility input and output statement contains the mnemonic Uio, which you code to identify your input and output devices.

In this mnemonic:

U
: Is a constant that identifies this as a utility input and output statement.

i
: Is a variable that identifies the input device as a card reader (C), magnetic tape unit (T), or disk or diskette (D).

o
: Is a variable that identifies the primary output device as a card punch (C), magnetic tape unit (T), disk or diskette (D), or printer (P).

Valid Uio mnemonics are:

Mnemonic	Input and Output Device
UCC	Card reader to card punch
UCD	Card reader to disk
UCT	Card reader to tape
UCP	Card reader to printer (Default option - when you copy a card file to the printer, and require no modifications to the file, no Uio mnemonic is needed.)
UDC	Disk to card punch
UDD	Disk to disk
UDT	Disk to tape
UDP	Disk to printer
UTC	Tape to card punch
UTD	Tape to disk
UTT	Tape to tape
UTP	Tape to printer

Figure 4-3 shows the Uio mnemonics and their relationship to the input and output devices needed to copy or compare files.

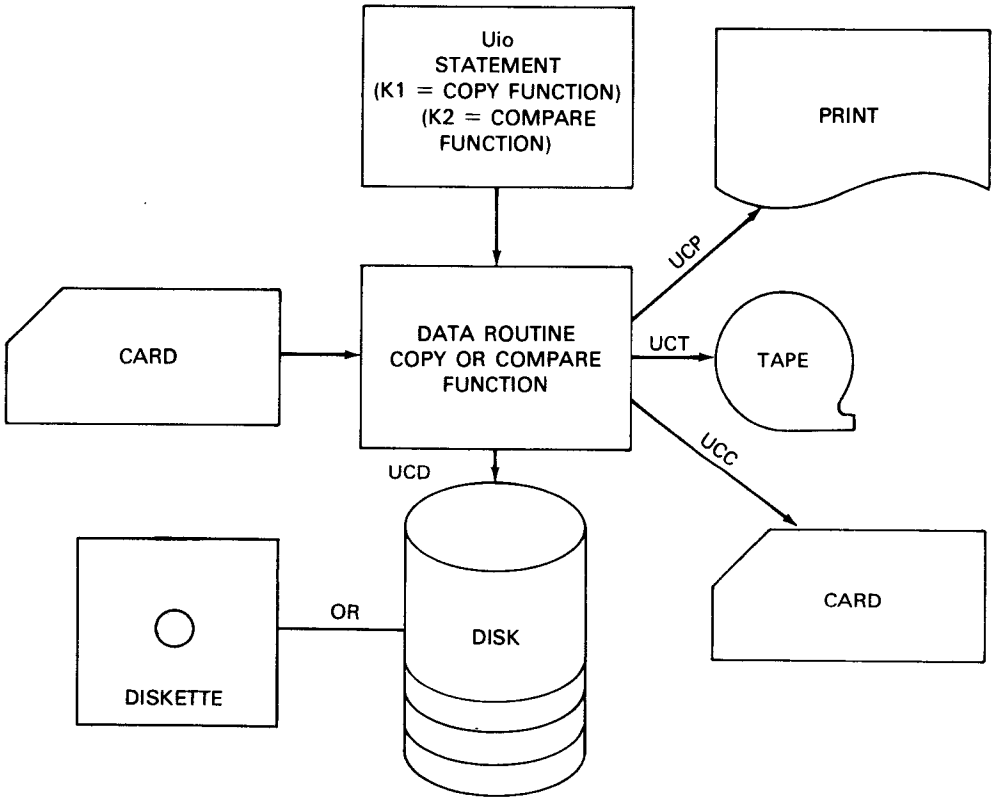


Figure 4-3. Relationship of Uio Mnemonics to Input and Output Devices (cont.)

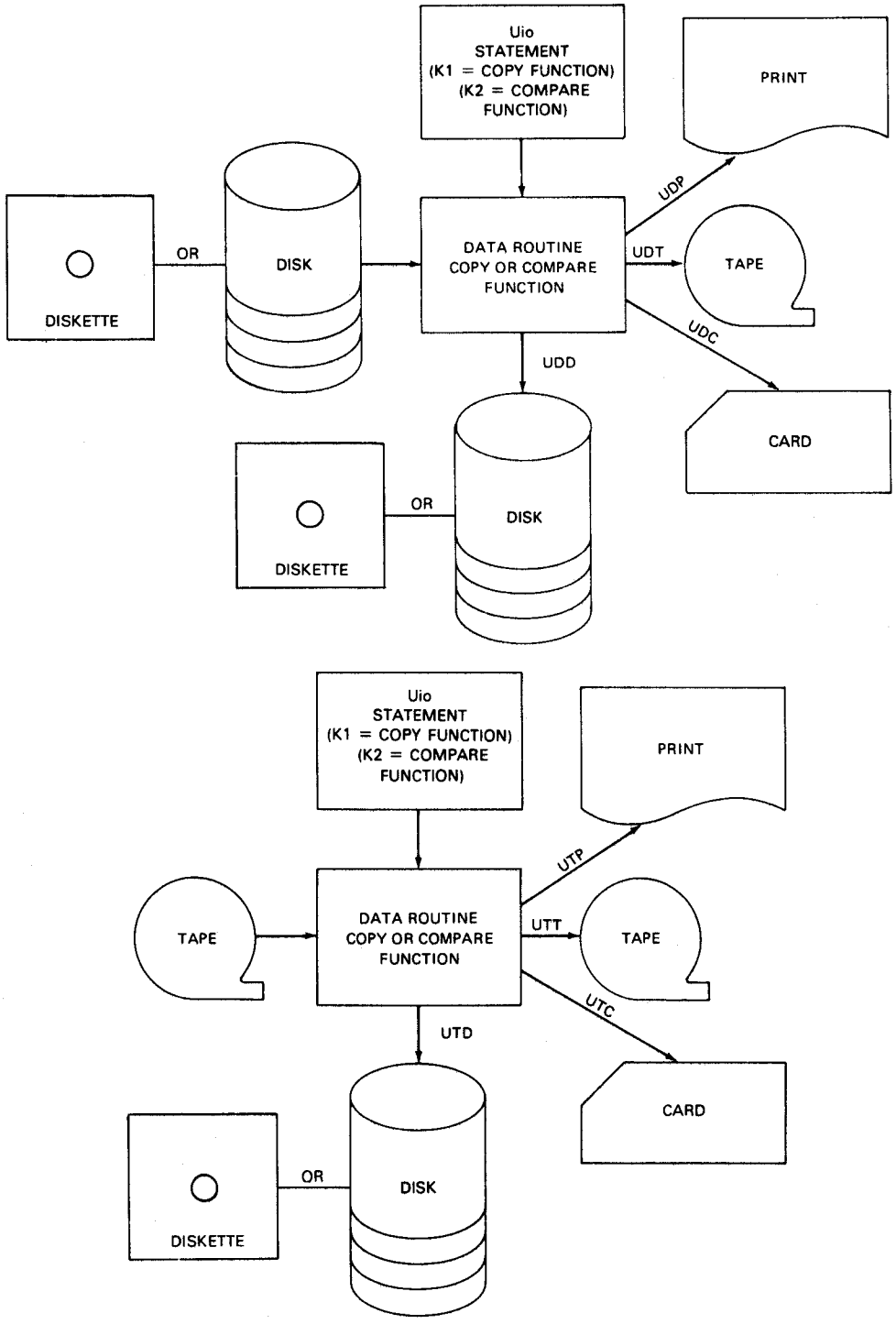


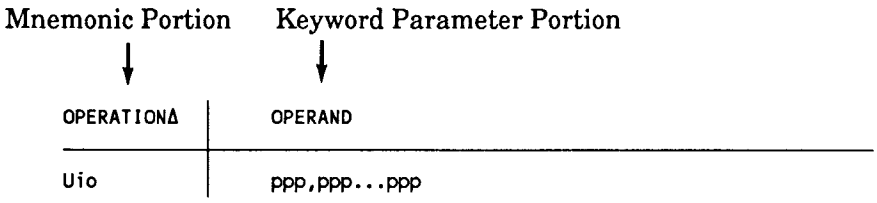
Figure 4-3. Relationship of Uio Mnemonics to Input and Output Devices

Uio Parameters

The available Uio parameters are required only when you want to perform an operation other than a file copying operation. Table 4-1 describes these parameters.

4.3.2. Input and Output Statement Formats

A number of different formats are available for the utility input and output (Uio) statements used to describe the types of files and devices that can be involved in DATA routines. These formats use variations of the same operand set, depending upon the device used to copy or compare files. The general format for each statement is:



Statements are written starting in column 1 and ending in or before column 71. The statements may be repeated as often as necessary; they may not be continued. If more than one card is required, do not insert a continuation character in column 72, but repeat the Uio operation in columns 1-3 of the next card.

The mnemonic portion, Uio, specifies the device type for the input (i) and output (o) files. It must be followed by a blank.

The keyword parameter portion, ppp, defines the files being used and the options that are needed. Each parameter is followed by a comma except the last parameter, which must be followed by a blank. Embedded blanks are not allowed except with alphanumeric literals. All keyword parameters are optional. Default values are supplied by the DATA routine.

The keyword parameters required by your job must be supplied in the Uio statement and in the form specified in Table 4-1. The complete Uio statement for your job must be written in one of the formats shown on the following pages. There is a format for every combination of input and output device types. The default options are indicated by shading.

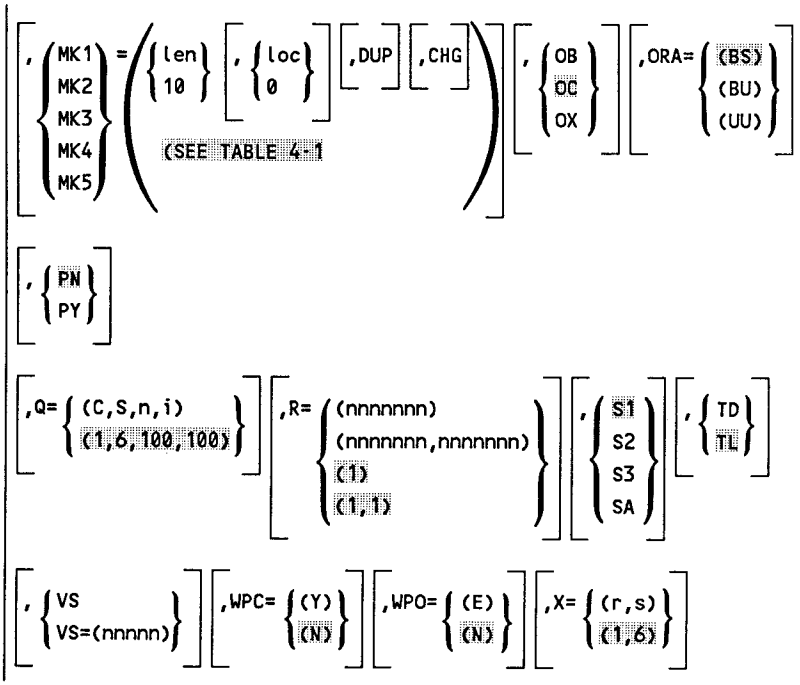
Card Input Formats

- Card-to-card

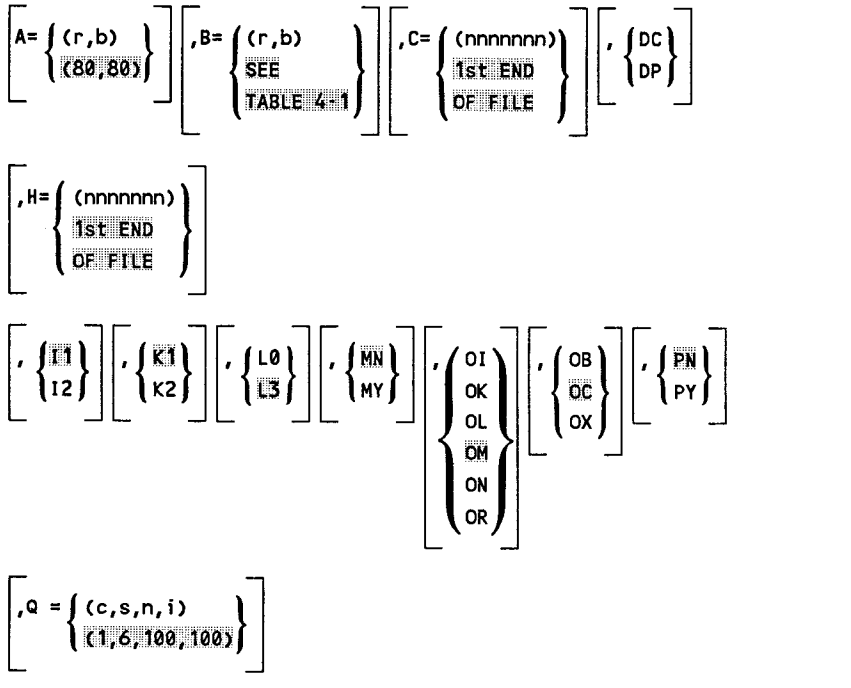
OPERATIONΔ	OPERAND
UCC	[A= {(r,b) / (80,80)}] [,B= {(r,b) / SEE / TABLE 4-1}] [,C=(nnnnnnn) / 1st END OF FILE] [,DP] [,H= {(nnnnnnn) / 1st END OF FILE}] [, {I1 / I2}] [, {K1 / K2}] [, {MN / MY}] [, {O1 / O2}] [, {OB / OC / OX}] [ORA= {(BS) / (BU) / (UU)}] [, {PN / PY}] [,Q= {(c,s,n,i) / (1,6,100,100)}] [,R= {(nnnnnnn) / (nnnnnnn,nnnnnnn) / (1) / (1,1)}] [, {S1 / S2 / S3 / SA}] [, {TD / TL}] [,WPC= {(Y) / (N)}] [,WPO= {(E) / (N)}] [,X= {(r,s) / (1,6)}]

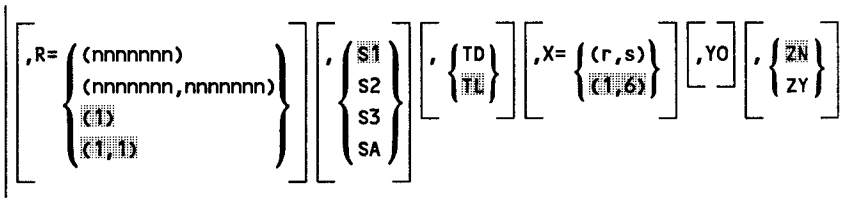
- Card-to-disk

OPERATIONΔ	OPERAND
UCD	[A= {(r,b) / (80,80)}] [,B= {(r,b) / SEE / TABLE 4-1}] [,C= {(nnnnnnn) / 1st END / OF FILE}] [, {DC / DP}] [,H= {(nnnnnnn) / 1st END / OF FILE}] [, {I1 / I2}] [, {K1 / K2}] [, {MN / MY}], {OM / OMY} = {(C[,V][,R]) / (I[,n][,V][,R])}

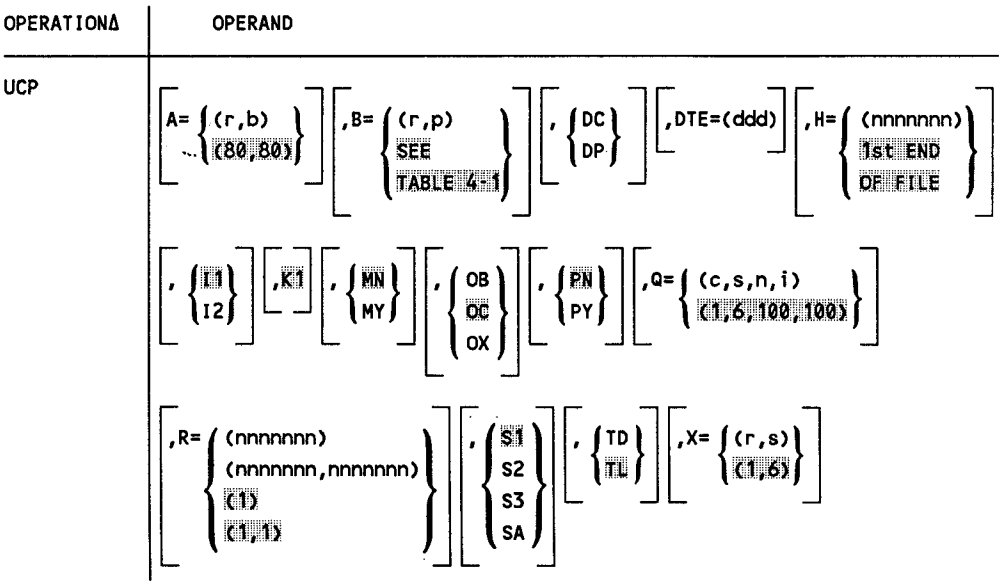


- Card-to-tape

OPERATIONΔ	OPERAND
UCT	



- Card-to-printer



Example 1

```
1        10        20        30        40        50        60        72
-------------------------------------------------------------------
UCT OR,Q=(2,4,1,1),R=(50),X=(2,4),ZN
```

A card reader and tape unit are the devices used in the operation designated by the UCT mnemonic. A file input in the card reader is copied into a magnetic tape output file.

- Default processing

  The default keyword parameters are omitted because their default values are automatically assigned to the job. Omitting the *A* and *B* keyword parameters specifies 80-byte record and block lengths. Fixed-length records are assumed. Omitting keyword parameters *I1* and *K1* specifies that card input is in EBCDIC and that this is a copy operation.

- Requested processing

  Options requested, via keyword parameters, to process the UCT statement include: *OR*, which specifies rewinding of output tape before and after processing; *Q=(2,4,1,1)*, which specifies that sequence numbers be written on the output tape file starting in column 2 (the sequence field is 4 bytes long, 1 is the first sequence number written, and that the sequence field is incremented by 1 for each record); *R=(50)*, *X=(2,4)*, and *ZN* specify that processing begins at logical record 50, a sequence check is made on the input file starting at column 2 for a length of 4 bytes, and leading tape marks are not written on the output file.

Example 2

```
1        10        20        30        40        50        60        72
-------------------------------------------------------------------------
UCC
```

A card reader and card punch are identified by the mnemonic UCC as the devices used. A file input in the card reader is copied into a card punch output file.

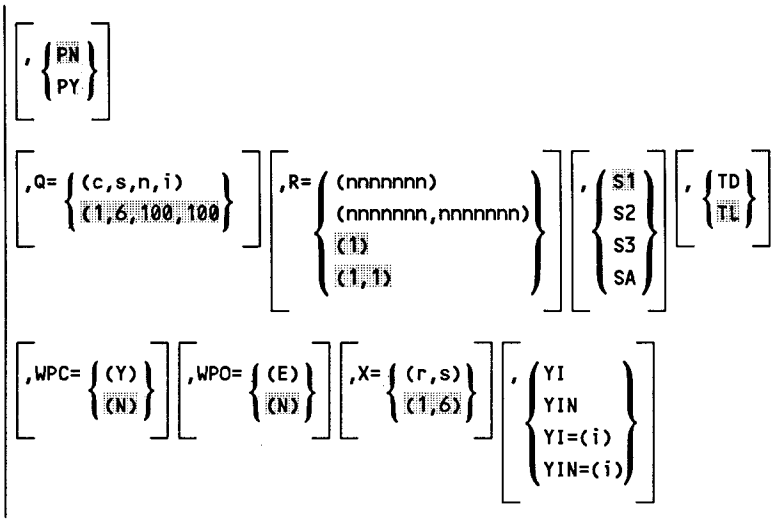
- Default processing

  Because the default values sufficiently describe the particular file and operation, all keyword parameters are omitted. The default values for the keyword parameters *A, B, I1, K1, O1,* and *R* specify that the input and output files contain 80-byte blocks and records, input mode is EBCDIC, this is a copy operation, output mode is EBCDIC, and processing begins with the first record.

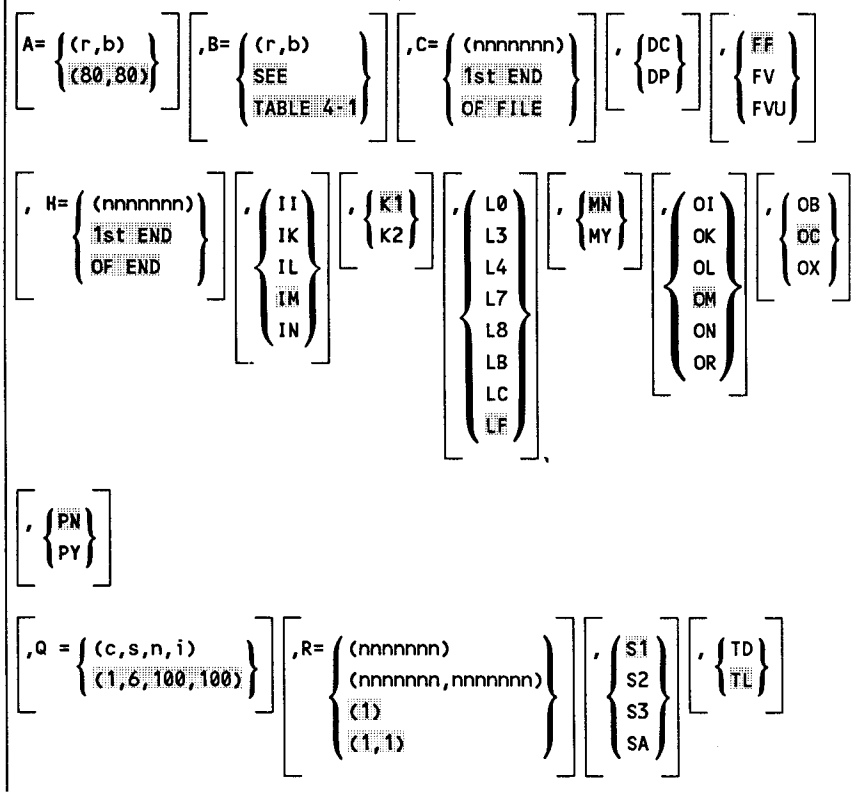
Tape Input Formats

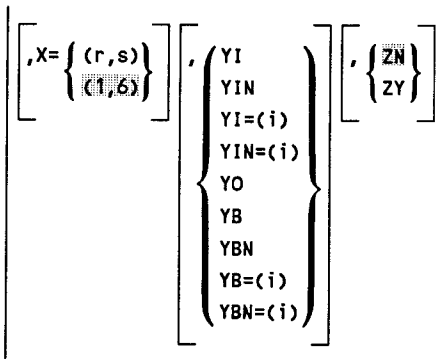
- Tape-to-card

OPERATIONΔ	OPERAND
UTC	[A= {(r,b) / (80,80)}] [,B= {(r,b) / SEE / TABLE 4-1}] [,C= {(nnnnnnn) / 1st END / OF FILE}] [,DP] [,H={(nnnnnnn) / 1st END / OF FILE}] [, {I1 / IK / IL / IM / IR}] [, {K1 / K2}] [, {L3 / L7 / LB / LF}] [, {MN / MY}] [, {O1 / O2}] [, {OB / OC / OX}] [,ORA= {(BS) / (BU) / (UU)}]



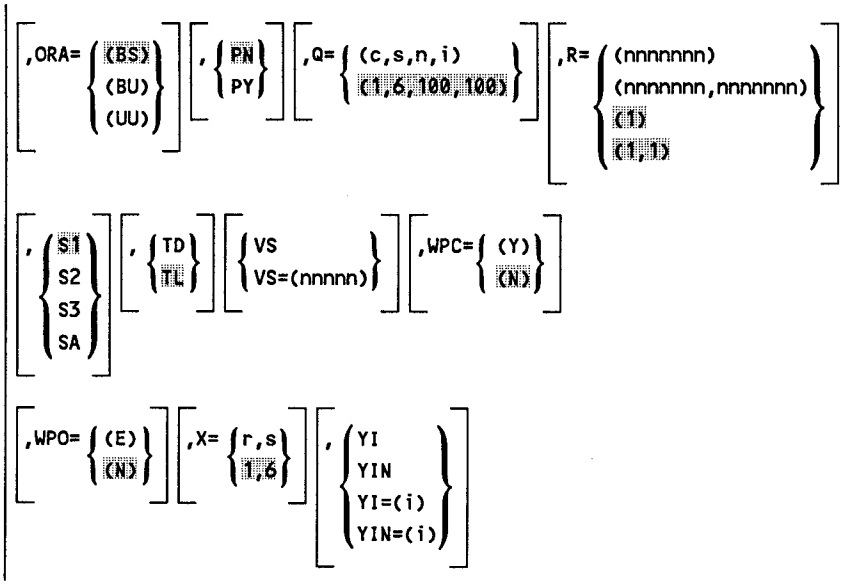
- Tape-to-tape

OPERATIONΔ	OPERAND
UTT	

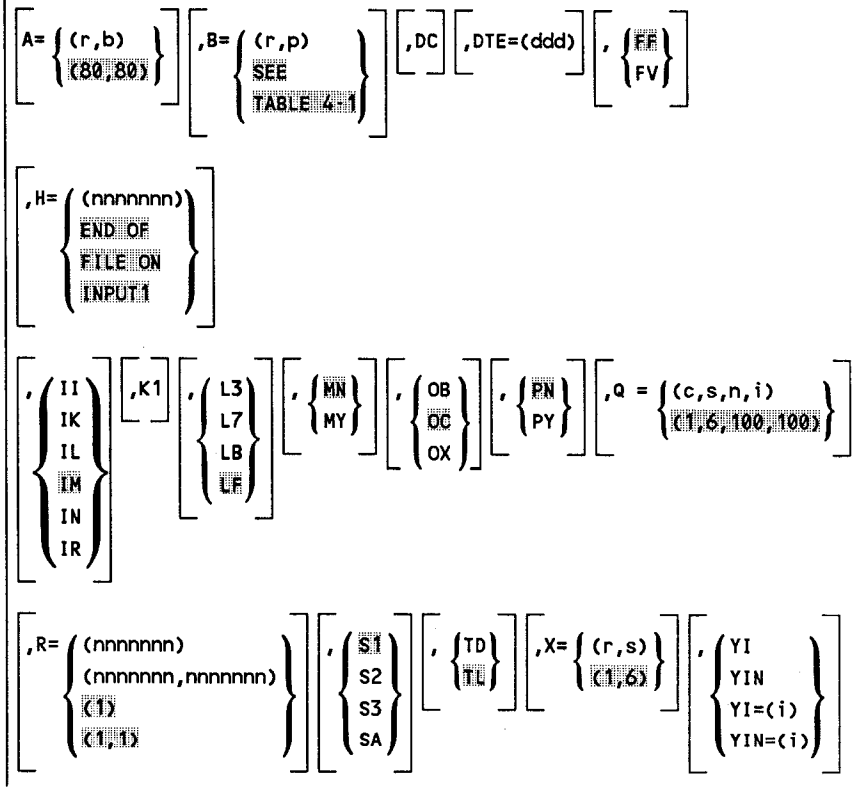


- Tape-to-disk

OPERATIONΔ	OPERAND
UTD	[A={(r,b) / (80,80)}][,B={(r,b) / SEE TABLE 4-1}][,C={(nnnnnnn) / 1st END OF FILE}][,{DC / DP}]
	[,{FF / FV=(nnnnn)}]
	[,H={(nnnnnnn) / 1st END OF END}][,{II / IR / IN / IK / IL / IM}][,{K1 / K2}][,{L0 / L4 / L8 / LC}][,{MN / MY}]
	[,{MK1 / MK2 / MK3 / MK4 / MK5}=({len / 10}[,{loc / 0}][,DUP][,CHG] / SEE TABLE 4-1)][,{OC / OX / OB}]
	[,{OM / OMY}={(C[,V][,R]) / (I[,{n / 1}][,V][,R])} / SEE TABLE 4-1]



- Tape-to-printer

OPERATIONΔ	OPERAND
UTP	

Example 1

```
1        10        20        30        40        50        60          72
-------------------------------------------------------------------------
UTT
```

A tape-to-tape operation is designated by the mnemonic UTT. The data on one magnetic tape file is to be copied to another magnetic tape file.

- Default processing

  Because the default options sufficiently describe the magnetic tape files and the operation, all keyword parameters can be omitted. The default keyword parameters, *A, B, FF, IM, K1, LF, OM, R,* and *ZN* specify that the input and output record and block lengths are 80 bytes. The input tape is not rewound before or after processing. This is a copy operation. There are no input or output file labels. The output tape is not rewound after processing. Processing begins with the first logical record; leading tape marks are not written on the output file.

Example 2

```
1        10        20        30        40        50        60          72
-------------------------------------------------------------------------
UTT A=(80,90),C=(500),IL,K2,OR,R=(5,5)
```

A tape-to-tape operation is designated by the mnemonic UTT. Two magnetic tape files are compared and the unequal records are printed.

- Default processing

  The *B* keyword parameter is omitted. By default it is assumed that the second input file block and record lengths are the same as the first input file specified by *A=(80,90)*. Omitting the *LF* keyword parameter indicates that the input and output files are unlabeled. By omitting the *ZN* keyword parameter, tape marks are not written on the output tape.

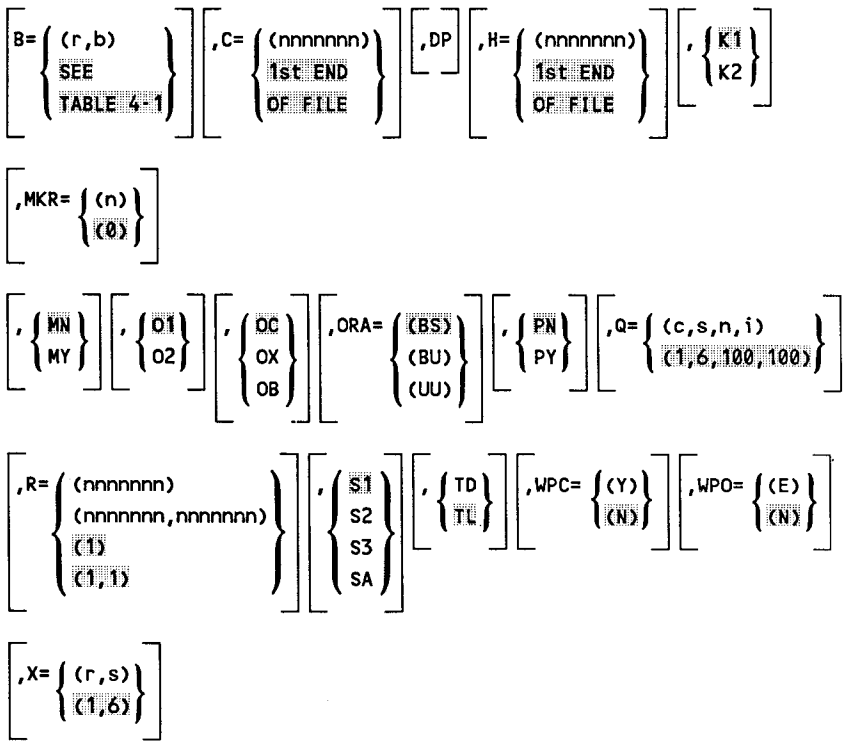
- Requested processing

  Input record lengths are 80 bytes long, and input block lengths are 90 bytes long as specified by the *A* keyword parameter. The *C* keyword parameter specifies that 500 unequal records are accepted and printed before job termination. The *IL* keyword parameter specifies that the first input tape is rewound before processing and rewound with interlock after processing. *K2* specifies a compare operation; *OR* specifies that the second input tape is rewound before and after processing. The *R* keyword parameter specifies that processing begins with the fifth logical record on both input tapes.

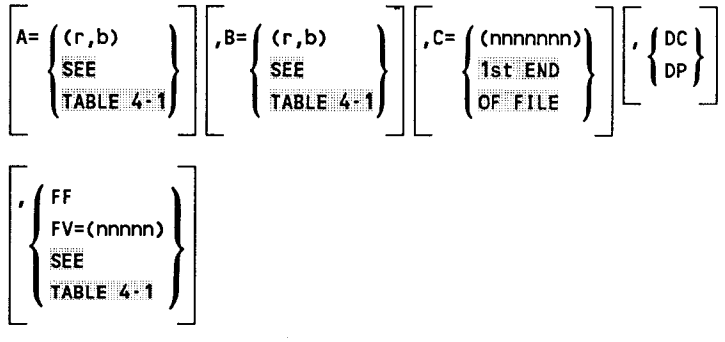
Disk Input Formats

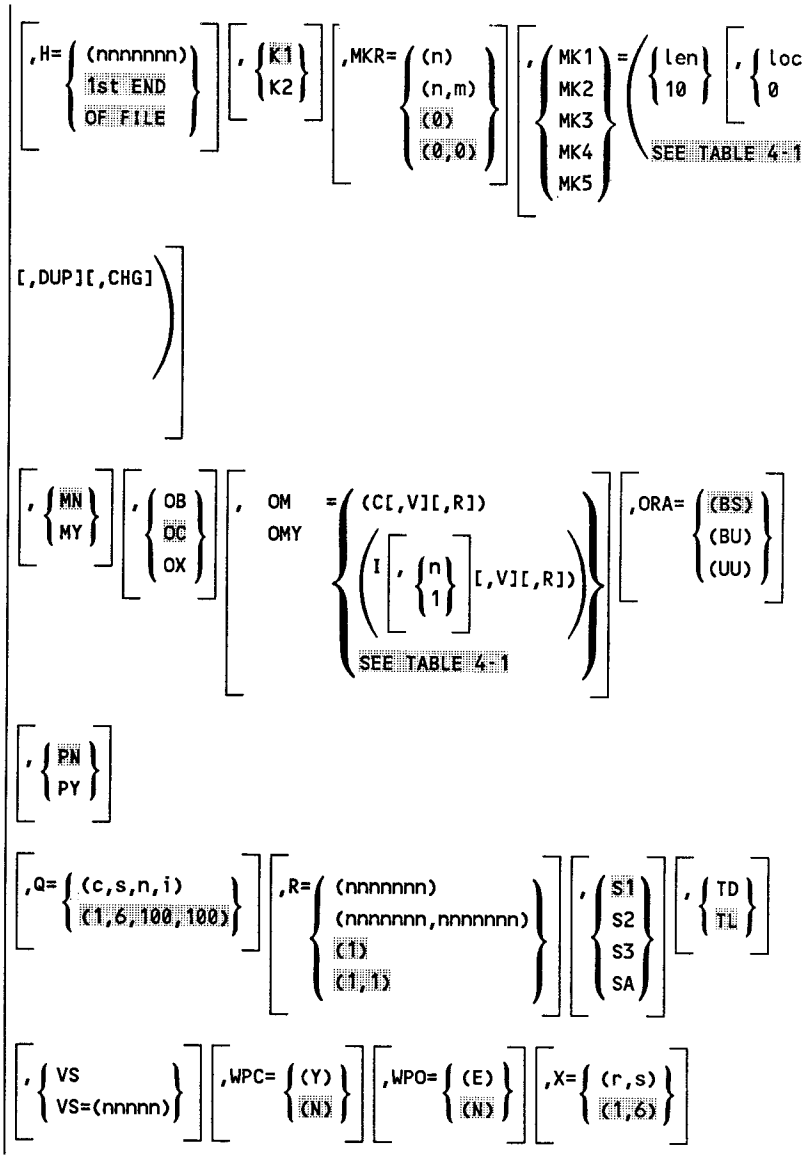
Diskettes are treated as disk files.

- Disk-to-card

OPERATIONΔ	OPERAND
UDC	

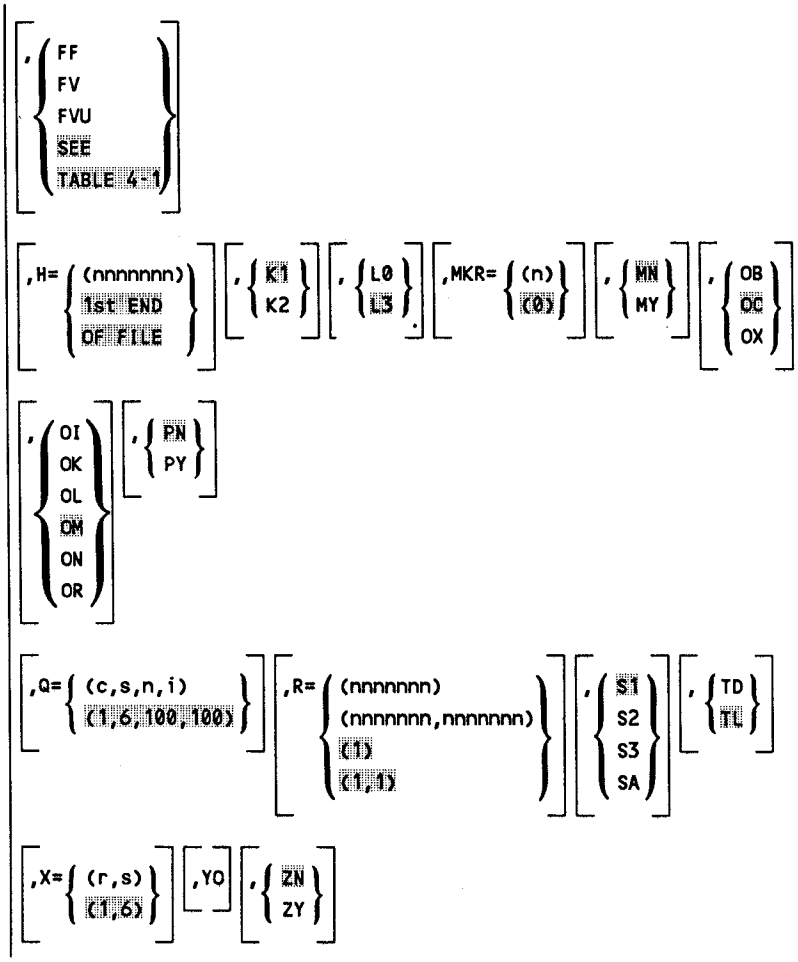
- Disk-to-disk

OPERATIONΔ	OPERAND
UDD	



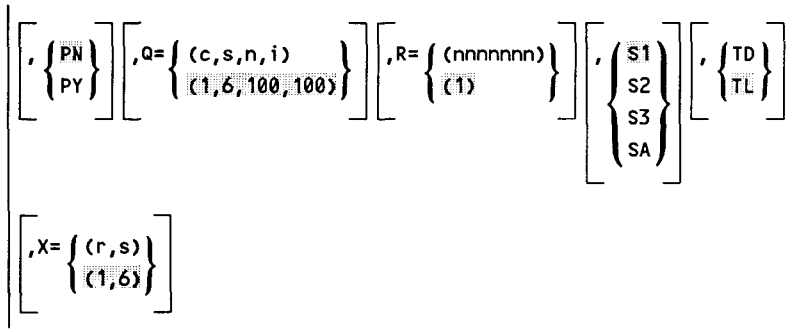
- Disk-to-tape

OPERATIONΔ	OPERAND
UDT	[A={(r,b) / SEE TABLE 4-1}][,B={(r,b) / SEE TABLE 4-1}][,C={(nnnnnnn) / 1st END / OF FILE}][,{DC / DP}]



- Disk-to-printer

OPERATIONΔ	OPERAND
UDP	[B={(r,p) / SEE TABLE 4-1}] [,DC] [,DTE=(ddd)] [,H={(nnnnnnn) / 1st END OF FILE}] [,K1] [,MKR={(n) / (0)}] [,{MN / MY}] [,{OB / OC / OX}]



Example 1

```
1        10        20        30        40        50        60          72
UDC C=(10),H=(1000),K2,R=(10,10)
```

The mnemonic UDC specifies a disk-to-card operation. An input disk file in MIRAM format is compared with the card file entered from the card reader. The unequal records are printed.

- Default processing

  All disk input file characteristics are taken from information stored in the VTOC. These include the length of the input records and blocks, the fixed-length record format, and the disk file which is in MIRAM format. Because the *O1* keyword parameter is omitted, EBCDIC card code is assumed. The length of the output records and blocks is equal to the length of the input records and blocks.

- Requested processing

  The requested keyword parameters specify the following: a maximum of 10 unequal records are accepted and printed; DATA routine is terminated after comparing 1000 records; this is a compare operation; and begin processing at the tenth record in the disk and card files.

Example 2

```
1        10        20        30        40        50        60          72
UDD
```

The mnemonic UDD specifies a disk-to-disk operation. Because no parameters are specified, the default values are used.

- Default processing

  All disk input file characteristics are taken from information stored in the VTOC. These include input and output file block and record lengths, fixed-length records, the key length, and the key location. By default, both printing and punching are suppressed. This is a copy operation. The output file type is the same as the input file type. No sequence checking is performed and processing begins with the first logical record. Both the input and output are not in ASCII code.

Input and Output Statement Keyword Parameters

To use the DATA routine, you should be thoroughly familiar with the keyword parameters listed in Table 4-1. They are listed in alphabetic order with a description of the function and default option for each. Also, MIRAM files are restricted to having key lengths less than or equal to 80 bytes.

Note: MIRAM files must have the volume mount indicator set to MULTI if the file is to be processed randomly later.

Table 4-1. Keyword Parameters for Utility Input and Output Statements

Keyword Parameters	Description
A=(r,b)	INPUT1 Record and Block Length For fixed length (FF) records on card or tape files, this parameter specifies record length (r) and block length (b) expressed in decimal. If omitted, r and b both default to 80. The block length b must be a multiple of r. For variable length (FV) records on tape input files, this parameter specifies a minimum r and a maximum b. If omitted, the minimum r defaults to 0 and the maximum b defaults to 80. For fixed length (FF) records, this parameter specifies a dummy value less than six digits long (r) and the buffer length (b) expressed in decimal (b must be a multiple of sector size). If the specified buffer length is less than the minimum allowed for the record size, as specified in the VTOC entry for INPUT1, then the minimum will be used.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
	For variable length (FV) records, this parameter specifies a minimum r and the buffer length (b must be a multiple of sector size). If the specified b is less than the minimum allowed for the record size, as specified in the VTOC entry for INPUT1, then the minimum is used. All subparameters to this parameter have a maximum size limit of 5 decimal digits.
B=(r,b)	OUTPUT1/INPUT2 Record and Block Length For fixed length (FF) records on card or tape files, this parameter specifies record length (r) and block length (b). For these files, b should be an even multiple of r. For fixed length (FF) records on disk files, this parameter specifies r (which includes key length(s) but excludes record control byte) and buffer length (b) which must be a multiple of sector size (sector size is normally 256 bytes). If omitted for fixed length (FF) records, r defaults to the INPUT1 file record length (from the A=( ) parameter for non-disk input or the file format labels for a disk input file). Block/buffer length (b) defaults to the INPUT1 file block/buffer length (from the A=( ) parameter for non-disk and input files or the file format labels for all other disk input files). For variable length (FV) records on disk files, this parameter specifies minimum r (which includes all keys plus the 4-byte variable record descriptor word) and b which must be a multiple of sector size (sector size is normally 256 bytes). If omitted for variable length (FV) records, the minimum r defaults to zero and the block/buffer length b defaults to the INPUT1 file block/buffer length. If comparing (K2) files and the INPUT2 file is a disk file, the record length for fixed record (FF) files and the block length for fixed (FF) and variable (FV) record files will be obtained from the disk file format labels and override the information in this parameter. For files, buffer size may be specified here. If it isn't, the minimum allowed is used. This is a performance consideration.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
B=(r,p)	For fixed length (FF) records on the primary printer file (UiP), this parameter specifies the record length (r) and the print line size (p). If this parameter is omitted in this case, r will default to the INPUT1 record length, and a 120-character print line size (p) will be assumed. If r is greater than the record length of the INPUT1 file, the OUTPUT1 record will be padded on the right with blanks. If the specified r is less than the record length of the INPUT1 file, the INPUT1 record length will be used. If first character forms control (SA) is specified, print line size (p) must include one byte for the control character. For variable length (FV) records on the primary printer file (UiP), this parameter specifies a minimum and a print line size (p). The minimum r cannot be less than 2. If this parameter is omitted in this case, the minimum r will default to the value used for the INPUT1 file, and a 120-character print line size will be assumed.
C=(nnnnnnn)	Compare Error Check This parameter can not be specified if copy (K1) is specified. It indicates the number (a decimal value represented by nnnnnnn) of unequal records accepted and printed prior to termination, which occurs when nnnnnnn+1 unequal records are found. If omitted, all unequal records are printed, and the job terminates on the first end-of-file.
DC	Dual Output File This parameter requests the creation of a dual card OUTPUT2 file. The first 80 bytes of each OUTPUT1 record will be punched. If the record length of the OUTPUT1 file is less than 80, the OUTPUT2 record will be blank-filled on the right. The device assigned to the OUTPUT2 file can be an 80 column card punch, a 96 column card punch. The OUTPUT2 records will be 80 bytes long regardless of the I/O device assigned to this file. This parameter can not be specified if file compare (K2) is specified, or if variable length record (FV) is specified, or if the OUTPUT1 file is a card file (UiC).
DP	This parameter requests the creation of a dual printer output file. Each record sent to the OUTPUT1 file is printed in its entirety. The print line size is fixed at 120 for this file. This parameter cannot be specified if file compare (K2), correction (COR), or primary printer file (UiP) is specified.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
DTE=(ddd)	Print Date on Output Listing This parameter prints the date on all output listings. The ddd specifies YMD in any order, where Y is the year, M is the month, and D is the day.
	Record Format
FF	Specifies fixed-length records.
FV	Specifies variable length records. This parameter can not be specified if any of the files assigned to the job are card files.
FV=(nnnnn)	Specifies variable-length records for files where n specifies the output file slot size for OUTPUT1 file. The FF AND FV parameters are mutually exclusive. If both of these parameters are omitted, and the INPUT1 file is a disk file (UDo), then the record format of the INPUT1 file is used. If both of these parameters are omitted, and the INPUT1 file is not a disk file, then fixed length (FF) is assumed.
FVU	Specifies variable-length records with unblocked records in the OUTPUT1 file. Note: Files having fixed-length records cannot be copied to files having variable-length records and vice versa.
H=(nnnnnnn)	Halt on Record Count Specifies the number of records to be processed before terminating. The term nnnnnnn represents a global count of all records in the file. If R=( ) is specified, the number of records to be skipped before processing are included in nnnnnnn for this parameter. The job terminates at first end of file.
	EBCDIC/Column Binary
I1	Specifies that the card INPUT1 file is in EBCDIC code

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
I2	This parameter indicates the card INPUT1 file is in column binary code. If this parameter is specified, then the INPUT1 record length, as specified in the first entry of the A=( ) parameter, should be twice as long as the number of characters in the INPUT1 record (column binary code requires two bytes to represent each character).
	Tape Input Rewind
II	Rewind before and rewind interlock after processing.
IK	No rewind before and rewind after processing.
IL	No rewind before and rewind with interlock after processing.
IM	No rewind before and no rewind after processing.
IN	Rewind before and no rewind after processing.
IR	Rewind before and after processing.
	If none of these parameters are specified, the INPUT1 file is not rewound before or after processing and IM is assumed.
	Copy and Compare
K1	This parameter requests a file copy operation. The INPUT1 file is copied to the OUTPUT1 file.
K2	This parameter requests a file compare operation. The INPUT1 file is compared to the INPUT2 file, and unequal records are printed. If this parameter is specified, a printer must be assigned to the job, but none of the printer file parameters can be specified. This parameter can not be specified if correction (COR) or if dual output (DP or DC) are specified.
	If both of these parameters are omitted, file copy (K1) is assumed.
	Tape Label
L0	Indicates standard labeled INPUT1 and standard labeled OUTPUT1/INPUT2.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
L3	Indicates standard labeled INPUT1 and unlabeled OUTPUT1/INPUT2.
L4	Indicates user labeled INPUT1 and standard labeled OUTPUT1/INPUT2.
L7	Indicates user labeled INPUT1 and unlabeled OUTPUT1/INPUT2.
L8	Indicates standard and user labeled INPUT1 and standard labeled OUTPUT1/INPUT2.
LB	Indicates standard and user labeled INPUT1 and unlabeled OUTPUT1/INPUT2.
LC	Indicates unlabeled INPUT1 and standard labeled OUTPUT1/INPUT2.
LF	Indicates unlabeled INPUT1 and unlabeled OUTPUT1/INPUT2. If all of these parameters are omitted, unlabeled INPUT1 and unlabeled OUTPUT1/INPUT2 (LF) are assumed.
	Input Key
MKR=(n)	This specifies the key of reference to be used for this access of the INPUT1 file where n is a one 1-digit decimal value between 0 and 5.
MKR=(n,m)	This specifies the key of reference to be used for this access of the INPUT1 file (n) and the key of reference to be used for this access of the INPUT2 file (m). Both n and m are 1-digit decimal values between 0 and 5. If 0 is specified in either of these parameters, the file can be either indexed or consecutive, but consecutive reads will be performed. If a non-zero value is specified in either of the above parameters, the file must be indexed, and a read by key will be performed using key (n) for INPUT1 and key (m) for INPUT2 as specified during the creation of INPUT1 and INPUT2, respectively. If these parameters are omitted, a read by relative record number will be performed.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
	Output Key
MKn=(len[,loc][,DUP][,CHG])	This specifies one of up to five output keys for an indexed or mixed OUTPUT1 file where: n Is a 1-digit number between 1 and 5 that specifies the key number. len Is a 2-digit number less than or equal to 80 that specifies the length of this key. loc Is a 5-digit number less than 32767 that specifies the key location, relative to zero, of this key. This will be the number of bytes that precedes the key. If this is omitted, 0 will be assumed. DUP Specifies that this key can have duplicates. If this is omitted, then no duplicates are allowed. CHG Specifies this key can be changed. If this is omitted, then no changes to this key are allowed. The keys specified by this parameter will be the only keys associated with this file. Any subsequent access by key must reference one of the keys defined by this parameter. If disk input and output are specified (UDD, OM parameter with valid specifications, and INPUT1 files), and the parameter is omitted, the OUTPUT1 file will be created with the same keys as specified in the VTOC entry for the INPUT1 file.
	Printer mismatch
MN	Do not ignore printer mismatches.
MY	Ignore printer mismatches.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
	If both of these parameters are omitted, MN is assumed. These parameters cannot be specified if compare (K2) or correction (COR) are specified.
	Output EBCDIC/Column Binary
O1	This indicates the OUTPUT1/INPUT2 card file is in EBCDIC code.
O2	This indicates the OUTPUT1/INPUT2 card file is in column binary code. If this parameter is specified, the OUTPUT1/INPUT2 record length, as specified in the first entry of the B=( ) parameter, should be twice as long as the number of characters in the OUTPUT1/INPUT2 record (i.e., column binary code requires two bytes to represent each character). If both of these parameters are omitted, EBCDIC (O1) is assumed.
	Tape Output Rewind
Ol	Rewind before and rewind interlock after processing.
OK	No rewind before and rewind after processing.
OL	No rewind before and rewind with interlock after processing.
OM	No rewind before and no rewind after processing.
ON	Rewind before and no rewind after processing.
OR	Rewind before and rewind after processing. If none of the parameters are specified, the OUTPUT1/INPUT2 tape is not rewound before or after processing and OM is assumed.
OM=(C[,V][,R]) or OM=(I[,nnn][,V][,R])	Specifies disk OUTPUT1 with no write verify, where: C Indicates a consecutive file. I Indicates an indexed file.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
	nnn Is a 3-digit decimal number from 1 through 127 specifying the number of 256-byte sectors the index buffer must accommodate. V Indicates that this file is to be treated as a multivolume file (whether or not it physically occupies more than one volume). A multivolume file can be processed randomly but cannot be extended. Leaving this indicator out causes the file to be treated as a single-volume file, which cannot be processed randomly but can be extended. When this indicator is omitted and input is disk, the DATA routine defaults to whatever value the input single-volume/multivolume file indicator is set to. When the input is card or tape, single-volume is always assumed for this indicator. Data-set-label diskettes are treated as card input, and single-volume mounting is assumed. Format-label diskettes are treated as disk input, and either mount option is available. R Indicates each record in the file is not to contain a record control byte.
OMY=(C[,V][,R]) or OMY=(I[,nnn][,V][,R]) or OMY=([,nnn][,V][,R])	Specifies disk OUTPUT1 with write verify. C, I, nnn, V, and R have the same meaning as they do for the OM=( ) parameter. If all of these parameters are omitted, a default will be assigned as follows: For a file compare (K2), where disk output was specified (UiD), the disk file type as specified in the VTOC entry for the INPUT2 file is used. For a file copy (K1), where disk input and disk output was specified (UDD), and none of the preceding parameters were specified, the OUTPUT1 disk file type will default to the INPUT1 disk file type.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
	Printer Mode
OC	Specifies output is printed in EBCDIC code.
OB	Specifies output is printed in both EBCDIC and hexadecimal code.
OX	Specifies output is printed in hexadecimal code.
	If all of these parameters are omitted, EBCDIC (OC) is assumed.
	Output Record Attribute for Diskettes
ORA=(BS)	Specifies that the output diskette records are to be blocked spanned.
ORA=(BU)	Specifies that the output diskette records are to be blocked unspanned.
ORA=(UU)	Specifies that the output diskette records are to be unblocked spanned.
	Specify UU for fixed-length, unblocked, unspanned records.
	If the ORA parameter is omitted, the default is BS; however, for fixed-length records, if sector size is 128 and record size is equal to or less than 128, the default is UU.
	Printer Numbering
PN	Specifies page number and date are not to be printed.
PY	Specifies page number and date are to be printed on the first line of each page. This parameter cannot be specified if first character forms control (SA) is specified.
	If both PN and PY are omitted, PN is assumed.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
	Sequence Numbering
Q=(c,s,n,i)	Indicates sequence numbering is to be done. c Is the column relative to column 1, where the sequence field begins. If omitted, 1 is assumed. s Is the size of the sequence field. If omitted, 6 is assumed. n Is the sequence number of the first record to be written (up to 10 decimal digits with or without leading zeros). If omitted, 100 is assumed. i Is the sequencing increment. If omitted, 100 is assumed. If the entire parameter is omitted, sequence numbering is not performed.
	File Repositioning
R=(nnnnnnn)	Indicates processing is to begin with logical record number nnnnnnn.
R=(nnnnnnnn,nnnnnnn)	For compare (K2) only. Specifies where processing begins in INPUT1 and INPUT2, respectively. The number of records skipped, as specified in this parameter, will be included in the record count for the H=( ) parameter, if it is specified. If R=( ) is omitted, processing begins with the first logical input record.
	Printer Spacing
S1	Single space after printing.
S2	Double space after printing.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
S3	Triple space after printing.
SA	The first character of each input record is used for forms control. This option is not allowed if the page numbering (PY) parameter is specified. If this parameter is specified, the device independent control characters must be used. See the *Consolidated Data Management Macroinstructions Programming Guide* (UP-9979) for details. If the SA parameter is specified, then TL and OC are assumed for print format and mode. These parameters can not be specified if correction (COR) or compare (K2) is specified. If these parameters are omitted, single spacing (S1) is assumed. The SA parameter is not allowed if a file compare (K2) is specified.
	Printer Format
TD	Print in display format.
TL	Print in list format. If TD and TL are omitted, list format (TL) is assumed. These parameters can not be specified if correction (COR) or compare (K2) is specified.
	Variable Sector Size
VS	This specifies that the data partition for an output file is to be written using a sector size other than 256 bytes. Also, it specifies that the DATA routine is to compute the sector size as being equal to the highest even multiple of the slot size that will fit into the buffer size as specified by the second entry in the B=( ) parameter. See the current version of the *Consolidated Data Management Macroinstructions Programming Guide* (UP-9979) for an explanation of slot size.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
VS=(nnnnn)	Specifies that the DATA routine is to use the value entered for nnnnn as the sector size. If both of these parameters are omitted, the sector size is 256 bytes.
	Write Protect of Output Diskette on Close
WPC=(N)	Specifies that the write protect option is to be left as specified.
WPC=(Y)	Specifies that the diskette be marked as write protected. If omitted, N is assumed.
	Write Protect of Output Diskette on Open
WPO=(E)	Erase the write protect.
WPO=(N)	Leave the write protect as specified. If you don't erase the write protect mark at open, data utilities cannot write to the file. If omitted, N is assumed.
	Sequence Checking
X=(r,s)	Specifies sequence checking, where r specifies the first column of the sequence field (relative to column 1), and s is the size of the sequence field (maximum value is 40). If no column number is specified, 1 is assumed; if no size is specified, 6 is assumed. The record is checked for a sequence w field that is higher than the corresponding field in the preceding record. When a sequence error occurs, an error message is logged, processing continues, and the new lower number is used as the new base for subsequent checking. If this parameter is omitted, sequence checking is not performed.
YB	Combines functions of parameters YI (ASCII input) and YO.
YBN	Combines functions of parameters YIN and YO.

continued

Table 4-1. Keyword Parameters for Utility Input and Output Statements (cont.)

Keyword Parameters	Description
YB=(i)	Combines functions of parameters YI=(i) and YO.
YBN=(i)	Combines functions of parameters YIN=(i) and YO.
	If all of the above parameters are omitted, EBCDIC output is assumed.
	ASCII Tape
YI	Indicates the INPUT1 tape file (UTo) will be in ASCII format. For fixed-length records (FF), a buffer offset length of 0 is used. For variable-length records (FV), a buffer offset length of 4 is used. A data management length check is performed using the contents of the buffer offset, which must contain the block length in decimal.
	Only variable-length ASCII output tapes are supported by data management, so only variable format is available for data utilities to create.
YIN	Identical to YI except that no length check is made for variable records.
YI=(i)	Identical to YI except that (i) is treated as a decimal value representing buffer offset length. For fixed-format records, the buffer offset is ignored. For variable-format records, the first four bytes are assumed to be the record length in decimal, and the rest of the buffer offset is ignored.
YIN=(i)	Identical to YI=(i), except that no length check is made for variable records.
	If any of the above parameters are omitted, EBCDIC input is assumed.
YO	Specifies ASCII output for variable-length record format; 4-byte buffer offset record length is written in decimal.
	Tape mark
ZN	Do not write tape mark on unlabeled output tape.
ZY	Write leading tape mark on unlabeled output tape.
	If both of these parameters are omitted, ZN is assumed.

4.3.3. Modifier Statements

You can modify DATA routine utility input and output (Uio) statements by using utility modifier statements:

- FS - Field Select Statement

  Move selected fields within a record

- SEL/DEL - Select/Delete Statements

  Select or delete records

- Hn - Title Statement

  Print page headings

- COR - Correction Statement

  Insert, delete, or replace records

If you use any of these optional statements, they must appear in your job control stream following the Uio statement. If no data utility modifier statements are used, the records are copied or compared without change.

The modifier statements FS, SEL/DEL, and COR have the same general format as the Uio statement (4.3.2). The statements are written starting in column 1 and ending in or before column 71. The keyword parameter portion (ppp) defines the operation and the files. Coding follows the same rules as for the Uio statement. The title (Hn) statement uses the format described in "Title Statement - Printing Page Headings" in this section.

Field Select Statement - Moving or Deleting Input Record Fields

The field select (FS) statement specifies the fields in the input records that are to be rearranged, omitted, or converted when copied from the input data record to an output record. The input fields must be within fixed-length records or within the fixed portion of a variable-length record. Fields may be moved to the same or different portions of the output record or deleted entirely from the output record. Fields in the output record not selected to receive data are blank-filled (X'40'). A maximum of 50 formatting options may be specified to move a record within the limits of available space required to store the information in main storage. If the output is to the printer, the list format must be used. (See Figure 2-5.)

The selected fields may be moved in the following ways:

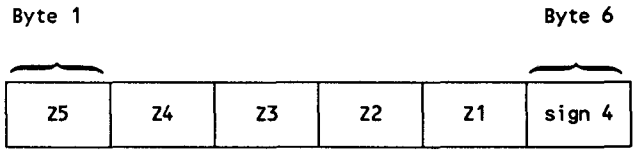
- Moved without change
- Converted from zoned to packed decimal and then moved
- Converted from packed to zoned decimal and then moved
- Converted to hexadecimal for printer output and then moved
- Converted, but remaining in the same location on the output record
- Deleted

Disk input key fields may be selected and transferred to output key fields or to data fields. However, the selected field must be contained entirely within either the disk key field or the data field.

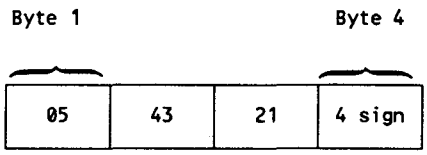
When converting a field from zoned to packed decimal, or packed to zoned decimal, the output field will be smaller or larger than the input field depending on the conversion you specify. The resulting field-selected record may not exceed the output record size, the output block size, or the printer line size as defined in your Uio statement. The rules that apply to the output field size are:

- Zoned to packed decimal conversion

  When the input field contains an even number of bytes, as in:



  The resulting output field is:

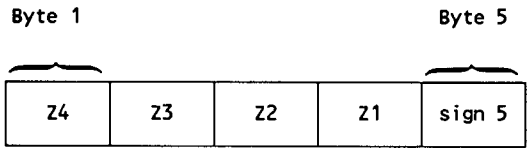


  To compute the size of the output field, apply the formula:

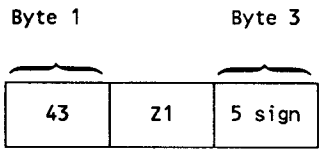
  $x=(n/2)+1$

  where n is the size of the input field and x is the size of the output field.

When the input field contains an odd number of bytes, as in:



The resulting output field is:



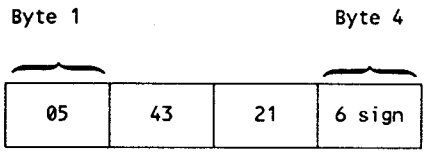
To compute the size of the output field, apply the formula:

$x=(n+1)/2$

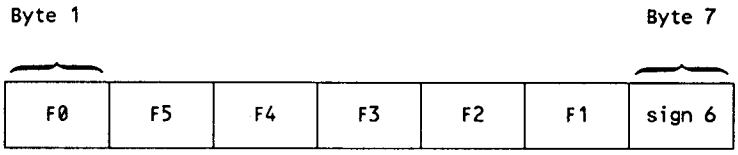
where n is the size of the input field and x is the size of the output field.

- Packed to zoned decimal conversion

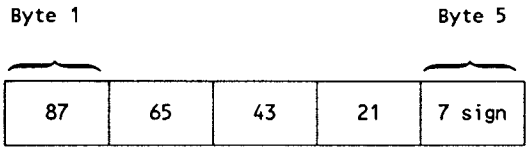
When the input field contains an even number of bytes, as in:



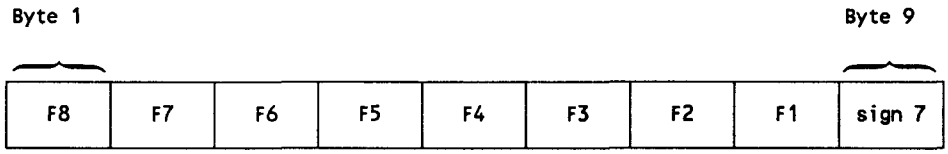
The resulting output field is:



When the input field contains an odd number of bytes, as in:



The resulting output field is:



Whether the input field contains an even or odd number of bytes, the size of the output field is computed by the formula:

$x=2n-1$

where n is the size of the input field and x is the size of the output field.

- Convert from EBCDIC character mode to hexadecimal

  The size of the output field is twice the size of the input field.

  Example

  EBCDIC input field

  0 1 2 3 4

  Hexadecimal output field

  F0F1F2F3F4

In the field select statement, all numbers are expressed in decimal and the specified starting position of the selected field is relative to the first character of the record (position 1). Commas separate definitions within each selected field; slashes separate the selected fields. Selected fields must be contained entirely on one card. Also, parentheses must be coded as shown.

There are two basic formats for the field select statement: one for fixed-length records and one for variable-length records.

FS Statement Format for Fixed-Length Records

OPERATIONΔ	OPERAND
FS	a,b,c[,d]/a,b,c[,d]/.../a,b,c[,d]

Each group of subparameters in the operand defines the field of a fixed-length input record and is specified in its output order. The subparameters are positional within each group.

a

Specifies the starting position of the fixed-length input record field.

b

Specifies the length of the field being copied or compared. The four options for this subparameter are:

$$\left\{\begin{matrix} s \\ (P,n,m) \\ (U,n,m) \\ (X,n) \end{matrix}\right\}$$

where:

s

Is a decimal number specifying the size of both the input and output field in bytes; this implies that no conversion is to take place.

P

Indicates that the contents of the input field are to be converted from zoned decimal to packed decimal before being transferred to the output file. The input field must be in EBCDIC.

U

Indicates that the contents of the input field are to be converted from packed decimal to zoned decimal (unpacked) before being transferred to the output file. The input field must be packed decimal.

X

Indicates that the contents of the input field are to be converted to hexadecimal before being transferred to the printer output file. The size of the output field must be twice the size of the input field. The input field must be packed decimal.

n

Is a decimal number specifying the byte size of the input field.

m

Is a decimal number specifying the byte size of the output field.

c

Specifies the starting position of the fixed-length output record field.

d

Is an optional parameter specifying that unpacked output fields are to reflect their sign. The four options are

P
: Print a plus (+) if the output field is positive.

M
: Print a minus (-) if the output field is negative.

B
: Print a plus (+) or minus (-) depending on whether the field is positive or negative.

N
: Print no sign.

If omitted, N is assumed.

FS Statement Format for Variable-Length Records

For variable-length records, the following rules for field selection apply:

- Only that portion of the input record that is fixed in size and is always present may be selected.
- The record length field is generated in the first four bytes of each output record. Selected fields may not be transferred into these four bytes.
- When the variable portion of a variable-length record is to be copied to output, the letters CV (copy variable) must be specified within the FS statement. The variable portion of the record is placed in the output record following the fixed portion of the record.

A maximum of 50 formatting options may be included. The definitions for the a, b, c, and d subparameters are given in "FS Statement Format for Fixed-Length Records" in this section. The various positions for the CV specification field are as follows:

OPERATIONΔ	OPERAND
FS	CV/a,b,c[,d]/a,b,c[,d]/a,b,c[,d]

The preceding format is used when the variable portion of the input record is to be moved to the output record before the selected fields are moved.

OPERATIONΔ	OPERAND
FS	a,b,c[,d]/a,b,c[,d]/a,b,c[,d]/CV

The preceding format is used when the variable portion of the input record is to be moved to the output record after selected fields have been moved.

OPERATIONΔ	OPERAND
FS	a,b,c[,d]/CV/a,b,c[,d]/a,b,c[,d]

The preceding format is used when the variable portion of the input record is to be moved to the output record between the moves of the selected fields.

FS Statement Examples

The following examples illustrate the use of the FS statement for both fixed-length and variable-length records.

Example 1. Pack fixed-length records

Assume that an input field record contains eight fields. The fields numbered 1, 2, 7, and 4 are to be moved, in that order, to the output area; fields numbered 2 and 7 are to be packed while being moved. The fields are:

Number	Field Name	Positions
1	Name	1-15
2	Hourly rate	6-20
3	Number of dependents	21-22
4	Earnings to date	23-30
5	Address	31-66
6	Date of service	67-71
7	Hours worked	72-74
8	Weekly earnings	75-80

The FS statement is:

```
1        10        20        30        40        50        60        72
FS 1,15,1/16,(P,5,3),16/72,(P,3,2),19/23,8,21
```

The resulting output record is:

Number	Field Name	Positions
1	Name	1-15
2	Hourly rate	16-18
3	Hours worked	19-20
4	Earnings to date	21-28

Example 2. Unpack and convert fixed-length records to hexadecimal

Assume that an input file record contains five fields. Fields numbered 1, 3, 4, and 5 are to be moved to output. Field 1 is to be unpacked. Field 3 is to be converted to hexadecimal. Fields 4 and 5 are to be moved without change. The input fields are:

Number	Field Name	Positions
1	Life number, packed format	1-5
2	Number of dependents	15-20
3	Name	21-40
4	Hours worked	62-65
5	Weekly earnings	66-70

The FS statement is:

```
1       10        20        30        40        50        60        72
FS 1,(U,5,9),1/21,(X,20),21/62,4,62/66,5,66
```

The resulting output record is:

Number	Field Name	Positions
1	Life number, unpacked format	1-9
2	Name, in hexadecimal	21-60
3	Hours worked	62-65
4	Weekly earnings	66-70

Example 3. Interchange fixed portion and copy variable portion of variable-length records

Assume that an input file contains variable-length records. The minimum length of a logical record is 24 bytes, and the maximum block length is 300 bytes. The fixed portion of the logical record is defined as 24 bytes consisting of two 10-byte fields and the 4-byte record-length field. The two 10-byte fields are to be interchanged, and the variable portion of each logical record is to be copied. The input fields are:

Number	Field Name	Positions
1	Field A	5-14
2	Field B	15-24
	Variable portion	25-275 (maximum)

The FS statement is:

```
1        10        20        30        40        50        60        72
FS 5,10,15/15,10,5/CV
```

The resulting output record is:

Number	Field Name	Positions
1	Field B	5-14
2	Field A	15-24
	Variable portion	25-275 (maximum)

Example 4. Move fixed portion of variable-length records

Assume that an input file contains variable-length records. The minimum length of a logical record is 34 bytes, and the maximum block length is 300 bytes. The fixed portion of the logical record is defined as four bytes and three 10-byte fields. The first and third 10-byte fields are to be moved to positions 5 through 24 in the fixed portion of the output record, and the second 10-byte field is to be moved to positions 35 through 44.

Number	Field Name	Positions
1	Record length	1-4
2	Name	5-14
3	Address	15-24
4	Life number	25-34
5	Variable portion	35-300 (maximum)

The FS statement is:

```
1        10        20        30        40        50        60        72
FS 5,10,5/25,10,15/CV/15,10,35
```

The resulting output record is:

Number	Field Name	Positions
1	Record length	1-4
2	Name	5-14
3	Life number	15-24
4	Blanks	25-34
5	Address	35-44
6	Variable portion	45-310 (maximum)

Select or Delete Statements - Selecting or Deleting Records

The select statements (SEL, SELAND, and SELOR) and the delete statements (DEL, DELAND, and DELOR) are used with the Uio statement to select or delete records, respectively, during a copy operation. The selection or deletion of records is based on matching a portion of each record with a search argument. The matching portion is defined as the search key field and is specified in the data portion of the record. The search argument consists of a relational operator and a character string that represents the constant against which all search fields are matched. The relational operators used to qualify the search argument are equal to (=), greater than (>), or less than (<).

The INPUT1 file is searched, comparing the specified search key field of each record with the search argument. If the search key field of an input record matches the search argument (depending on the relational operator), then the record is either selected or deleted, depending on the requested function. Either SEL or DEL may be specified, but not both. Only one SEL or one DEL statement can be specified. If more than one is entered during a job, only the last SEL or DEL statement is executed. You can specify a maximum of five SELAND, SELOR, DELAND, or DELOR statements, each containing a single argument; however, the statements must all be the same (that is, all SELAND statements or all DELOR statements).

If the FS statement is also specified, the select or delete function is performed before the field select function.

The format for the select and delete statements is:

OPERATIONΔ	OPERAND
SEL SELAND SELOR DEL DELAND DELOR	D=(nnnnn),A{ > / = / < }{(sssss,aa...a) / (sssss,X'bbbb...bb'}

SEL is the operation code used to select records when a match exists between the search key field and the search argument. All records not selected are ignored.

DEL is the operation code used to delete records when a match exists between the search key field and the search argument. All records not deleted are used as input.

SELAND and DELAND are the operation codes for the "select and" and "delete and" functions, which allow from one to five search key field/search argument pairs. If *all* of the search key fields of an input record match their corresponding search arguments, the record is selected or deleted for output.

SELOR and DELOR are the operation codes for the "select or" and "delete or" functions, which allow from one to five search key field/search argument pairs. If *any* of the search key fields of an input record match their corresponding search arguments, the record is selected or deleted for output.

Keyword parameter *D=(nnnnn)* specifies, in decimal, the starting position of the first byte of the search key field relative to the first character in the record. If the record is a disk record with a key field, this parameter indicates the position of the search key field in the data portion of the record.

Keyword parameter *A* is the search argument definition, where *sssss* is the length in bytes of the search argument and *aa...a* or X'*bbbb...bb*' is the search argument. The value of the comparison for a match between the search key field and the constant in the search argument is indicated as follows:

A= {(sssss,aa...a) / (sssss,X'bbbb...bb')}

All records whose search key field is equal to the search argument are either selected or deleted depending on the function specified.

A> {(sssss,aa...a) / (sssss,X'bbbb...bb')}

All records whose search key field is greater than the search argument are either selected or deleted depending on the function specified.

A< {(sssss,aa...a) / (sssss,X'bbbb...bb')}

All records whose search key field is less than the search argument are either selected or deleted depending on the function specified.

Subparameter *sssss* specifies the length of the search argument in bytes (maximum length = 48 bytes, where one byte equals one character).

Subparameter *aa...a* is the character string representing the search argument specified as alphanumeric characters.

Subparameter X'*bbbb...bb*' is the character string representing the search argument specified as pairs of hexadecimal digits.

Example 1

```
1        10        20        30        40        50        60          72
-------------------------------------------------------------------------
 SEL D=(6),A=(3,500)
```

The search field for each record starts in position 6 of the data portion of the record, and is matched for an equal condition with the search argument containing a 3-byte character string of 500. The records that satisfy the match are written to the output device.

Example 2

Suppose you want to select all records for women older than 35. If the sex of the person is in column 3 and the age is in column 50 of the record, the following SELAND statements are used:

```
1        10        20        30        40        50        60          72
-------------------------------------------------------------------------
 SELAND D=(3),A=(1,F)
 SELAND D=(50),A>(2,35)
```

Title Statement - Printing Page Headings

The title (Hn) statement is used, in conjunction with those Uio statements that specify the printer as the primary or secondary output device, to print page headings. You can code a maximum of three Hn statements (each with a maximum of 160 characters) to have page headings printed on the first three lines of each page. If you have included the *PY* keyword parameter in your Uio statement, page headings begin on the first line of each page. The title statement is not supported for compare (K2) operations.

The Hn statement has the following format:

```
    1        10        20        30        40        50        60          72
    |--------------------------------------------------------------------------
1.  | Δ Hn<---------->heading<-------------------------------------------------->
2.  | Δ  <---------------->heading<-------------------------------------------->X
3.  | Δ  <------>heading<------->
```

where:

n
: Is 1, 2, or 3 specifying that the character string beginning in column 5 is to be printed as line 1, 2, or 3, respectively, of the title.

heading
: Is the character string representing the title. This may be a maximum of 160 characters, with 66 characters coded on line 1, 70 characters on line 2, and 24 characters on line 3.

X
: Represents any nonblank character used to indicate continuation. The continuation character is coded in column 72. The heading is continued on the next line (2 or 3) beginning in column 2. The DATA routine assumes that column 2 of line 2 is location 67 of the heading and that column 2 of line 3 is location 136 of the heading.

When the DATA routine scans your Hn statement, it assumes column 5 of the Hn statement is location 1 of the heading; therefore, a character to be printed in position n of a heading must be punched in column n+4 of the Hn statement. For example, if a particular column of information in your job is to be printed beginning in location 10, you must code the heading in the Hn statement beginning in column 14.

If you define a printer line length via the B=(r,p) keyword parameter in the UCP, UDP, or UTP statement, the character string must not exceed this length.

Example

```
     1        10        20        30        40        50        60          72
1.  | Δ H1          GOLF TOURNAMENT
2.  | Δ H2    PLAYER                SCORES
```

Printer output:

GOLF TOURNAMENT

PLAYER SCORES

The printed output of line 1 shows the character string GOLF TOURNAMENT beginning in print position 10; PLAYER beginning in print position 5, and SCORES beginning in print position 24. These character strings were coded four positions to the right because the DATA routine considers column 5 in the Hn statement equal to print position 1 on the printed page.

Correction Statement - Correcting Records

The correction (COR) statement is used to correct records in an existing file during a copy operation. The corrections are made by means of a series of control cards. These control cards are read in as data cards from the control stream. Through their use, a file can have records inserted, deleted, or replaced. The control cards must be in numerical order according to record number. The corrections appear on the OUTPUT1 file; the records in the INPUT1 file are unchanged. In the event of deletions or insertions, the records may be reblocked to suit record positions.

The COR statement specifies the number of new correction records to be processed, where processing is to begin in the file, and where processing is to end, if applicable.

When replacing or inserting records, data cards containing the new or corrected data must immediately follow the COR statement that describes the location in the file for these records. The COR statement, together with any associated data cards, must be the last utility modifier statement in your control stream.

The format of the COR statement is:

OPERATIONΔ	OPERAND
COR	N=(x),A=(s),B=(e)

Keyword Parameter N

N=(x)
Specifies the number of correction records (99 maximum) to be replaced or inserted. If the records are to be deleted, *x* must be zero.

Keyword Parameter A

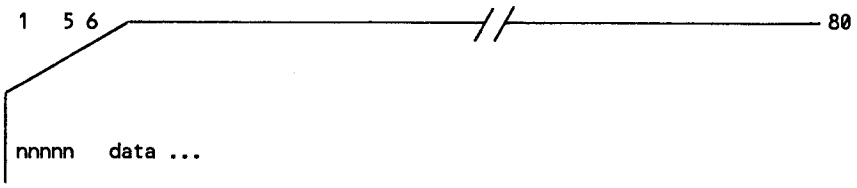
A=(s)
Specifies the 1- to 7-digit number of the starting record within the file. When replacing or deleting records, *s* specifies the record number of the starting location for the correction records. When inserting corrections, s specifies the record number that the correction records are to follow.

Keyword Parameter B

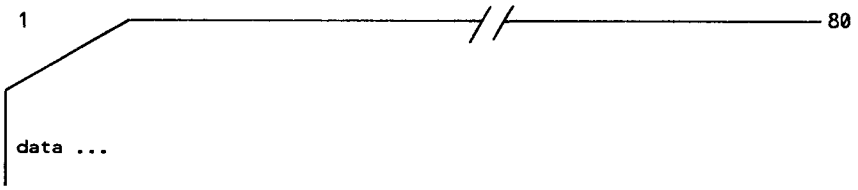
B=(e)
Specifies the 1- to 7-digit number of the ending record within the file. The ending record, specified by *e*, is the last one to be replaced by the new records.

The data card formats that follow the COR statements are

First data card:



Subsequent cards:



Columns 1 through 5 of the first data card specify the number of bytes supplied on the data cards for this record. For variable record format INPUT1 files, the DATA routine will construct a 2- or 4-byte record descriptor word as appropriate for the INPUT1 file type if field selection is specified or the OUTPUT1/OUTPUT2 file type if field selection is not specified. The maximum length is 32767. Columns 6 through 80 of the first data card contain positions 1 through 75 of the record to be added. If the record length is less than 75, the rightmost columns of the card are ignored.

If the data exceeds column 80 of the first data card, subsequent cards may be used with the data description beginning in column 1. As many cards as necessary can be used to supply the required number of characters in the record. The subsequent cards successively contain positions 76 through 155, 156 through 235, 236 through 315, etc., of the record. Portions of the record that exceed the length specified by nnnnn are ignored.

Each new record must begin on a new card.

Example

```
    1        10        20        30        40        50        60        72
1.  | COR N=(2),A=(1),B=(2)
2.  | 00017TOURNAMENTΔRECORD
3.  | 00020NOΔTHREEΔPUTTΔGREENS
4.  |  COR N=(2),A=(6)
5.  | 00013AVERAGEΔDRIVE
6.  | 00009260ΔYARDS
7.  |  COR N=(0),A=(10),B=(12)
```

The first COR statement specifies that the contents of lines 2 and 3 are to be placed between record 1 and record 2. The COR statement in line 4 specifies that two records are to be inserted after record 6. Lines 5 and 6 contain the data to be inserted. The last COR statement specifies that all records thru record 10 and record 12 are to be deleted.

Section 5
Control Streams

5.1. Purpose and Application

The sample programs in this section illustrate some of the functions of the DATA routine. These programs use the same input data and represent a typical progression for handling data.

Selected fields from the input data card deck are written to a work area on disk and then the two are compared. The data in the work area is copied to another work area on the disk and again these two are compared. The data in the first work area is printed out. Finally, one of the records in the data file is corrected.

The input data deck shown is for illustrative purposes; actual data decks could be considerably larger. All card and disk files used in these examples are in MIRAM format. Job control statements, Uio statements, utility modifier statements, and data decks needed for the job steps are included.

The amount of minimum main storage required for each job step is computed according to the specifications shown in 2.3.1.

5.2. Copy Card-to-Disk Operation

In this example the employee records of the data deck are copied to a disk file and written to the printer file. Some fields of the input file are rearranged by the FS statement, and records are deleted by the DEL statement. Also requested is the printing of headings on the printer output.

Job control stream

```
                    1        10        20        30        40        50        60        72
                    |--------|---------|---------|---------|---------|---------|---------|
                 1. // JOB DATAUT,,A000
                 2. // DVC 20      // LFD PRNTR
JOB              3. // DVC 30      // LFD INPUT1
CONTROL          4. // DVC 51      // VOL DSP028
CARDS            5. // EXT MI,C,,CYL,10
                 6. // LBL WORK2  // LFD OUTPUT1,,INIT
                 7. // EXEC DATA
                 8. /$
Uio CARD         9. UCD B=(90,512),DP,MY,PY,S2
```

```
                10.  FS 1,10,25/15,20,1/45,3,45/55,2,55/65,3,65/78,3,78/1,10,81
UTILITY         11.  DEL D=(1),A=(4,1111)
MODIFIER        12.   H1                          EMP            HOURLY    EMP        EMPX
CARDS           13.             SEQ
                14.   H2        NAME              NO             RATE      CLASS      RATIX
                15.   NG         NO
                16.  /*
                17.  /&
                18.  // FIN
                19.  // DATA FILEID=DATAUTINPUT1
                20.  0000789123    DONAHUE, MARY M          255       10        030        120
                21.  0000776312    GIAGLONE, JOHN K         255       10        035        125
                22.  0000617892    LANGINES, SAMANTHA R     270       11        040        130
                23.  1111777321    LONG, GEORGE A           255       10        063        135
                24.  0000678978    LOTIERZO, HELENE S       310       13        050        140
DATA            25.  0000776321    MATHEWS, SAMUEL T        425       15        071        145
CARDS           26.  0000667843    MATTHEWS,JANE L          270       11        045        150
                27.  0000667712    NORDICK, JOHN M          425       14        055        155
                28.  1111573212    OGALSBY, PETER K         500       15        075        160
                29.  0000673124    PARYIN, THOMAS G         425       14        060        165
                30.  /*
                31.  // FIN
```

Printer output

```
PAGE  0001   DATE 75/02/11

        NAME                EMP NO      HOURLY RATE   EMP CLASS   EMP RATING   SEQ NO

DONAHUE, MARY M          0000789123        255          10          030        1200000789123
GIAGLONE, JOHN K         0000776312        270          11          035        1250000776312
LANGINES, SAMANTHA R     0000617892        270          11          040        1300000617892
LOTIERZO, HELENE S       0000678978        310          13          050        1400000678978
MATHEWS, SAMUEL T        0000776321        425          15          071        1450000776321
MATTHEWS, JANE L         0000667843        270          11          045        1500000667843
NORDICK, JOHN M          0000667712        425          14          055        1550000667712
PARYIN, THOMAS G         0000673124        425          14          060        1650000673124
```

The job control cards which assign devices, allocate main storage and disk space, and initiate the DATA utility routine are included in the job control stream (cards 1 through 8). The printer file, card reader file and disk file are assigned, plus the space required on the disk file (card 5). If space had been allocated previously to the WORK2 label of the disk file (card 6), the // EXT job control statement would not be required (card 5).

The UCD mnemonic of the Uio statement specifies that this is a card-to-disk operation. The *A* keyword parameter is not coded because the default value of 80-byte record and block lengths applies. The *B* keyword parameter specifies the output record and block length of 90 bytes. The *DP* keyword parameter specifies printing of output records.

The keyword parameter *K1* is not coded because the default values apply to this job step. *K1* specifies a copy operation.

Keyword parameters *MY, PY,* and *S2* specify ignore printer mismatches, print page numbers and date on the first line of each page, and double-space the printer file output. The printer output is written in the *list* format by the default value of the *TL* keyword parameter.

The fields beginning in columns 1 and 15 of the card file are moved to different locations in the disk file by the specifications in the first two parameters of the FS statement. The first field of the card file is repeated in columns 81 through 90 of each record in the output disk file by the specifications in the last parameter (card 10).

The DEL statement (card 11) deletes all records containing the characters 1111 in the first four columns of the card file. Therefore, the input data records in cards 23 and 28 are not copied to the disk file and do not appear in the printer output.

Note that the column headings in the Hn statements (cards 12 through 15) are coded five positions to the right of their positions shown on the printer output. Within the DATA utility routine, column 5 of Hn statements is considered equal to column 1 ("Title Statement-Printing Page Headings" in Section 4).

The continuation of the title lines is shown in cards 13 and 15.

The // DATA FILEID job control statement on card 19 directs the card reader input to a spool file for the job. DATAUTINPUT1 is a concatenation of the job name and the LFD file name for the card reader. To spool in the data, key in the IN command at the console.

The printer output shows that all DATA utility routine specifications have been executed successfully. In addition, the termination information printout is produced showing the input and output file characteristics. (See 2.3.3 for a typical example.)

5.3. Compare Card-to-Disk Operation

This example illustrates the programmer messages that result when unequal records are compared.

Job control stream

```
                1        10        20        30        40        50        60        70        80
                |--------|---------|---------|---------|---------|---------|---------|---------|
                1.  // JOB DATAUTIL,,A000
                2.  // DVC 20     // LFD PRNTR
JOB             3.  // DVC 30     // LFD INPUT1
CONTROL         4.  // DVC 51     // VOL DSP028
CARDS           5.  // LBL WORK2  // LFD INPUT2
                6.  // EXEC DATA
                7.  /$
Uio CARD        8.  UCD B=(90,512),K2
                9.  /*
                10. /&
                11. // FIN
                12. // DATA FILEID=DATAUTILINPUT1
                13. 0000789123    DONAHUE, MARY M          255       10        030       120
                14. 0000776312    GIAGLONE, JOHN K         255       10        035       125
                15. 0000617892    LANGINES, SAMANTHA R     270       11        040       130
DATA            16. 1111777321    LONG, GEORGE A           255       10        063       135
CARDS           17. 0000678978    LOTIERZO, HELENE S       310       13        050       140
                18. 0000776321    MATHEWS, SAMUEL T        425       15        071       145
                19. 0000667843    MATTHEWS, JANE L         270       11        045       150
                20. 0000667712    NORDICK, JOHN M          425       14        055       155
                21. 1111573212    OGALSBY, PETER K         500       15        075       160
                22. 0000673124    PARYIN, THOMAS G         425       14        060       165
                23. /*
```

Printer output

```
INPUT1
0000789123    DONAHUE, MARY M              255       10        030       120
INPUT2
DONAHUE, MARY M          0000789123        255       10        030       1200000789123
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000776312    GIAGLONE, JOHN K             255       10        035       125
INPUT2
GIAGLONE, JOHN K         0000776312        255       10        035       1250000776312
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000617892    LANGINES, SAMANTHA R         270       11        040       130
INPUT2
LANGINES, SAMANTHA R     0000617892        270       11        040       1300000617892
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
1111777321    LONG, GEORGE A               255       10        063       135
INPUT2
LOTIERZO, HELENE S       0000678978        310       13        050       1400000678978
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000678978    LOTIERZO, HELENE S           310       13        050       140
INPUT2
MATHEWS, SAMUEL T        0000776321        425       15        071       1450000776321
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000776321    MATHEWS, SAMUEL T            425       15        071       145
INPUT2
MATTHEWS, JANE L         0000667843        270       11        045       1500000667843
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000667843    MATTHEWS, JANE L             270       11        045       150
INPUT2
NORDICK, JOHN M          0000667712        425       14        055       1550000667712
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
INPUT1
0000667712    NORDICK, JOHN M              425       14        055       155
INPUT2
PARYIN, THOMAS G         0000673124        425       14        060       1650000673124
ERROR DETECTED IN COMPARE OF FILE1 TO FILE2
UDD14 INPUT1 AND INPUT2 ARE OF UNEQUAL LENGTH
```

We used our original data deck (cards 13 through 22) to compare the card file to the WORK2 area of the disk file. The job control cards assign the printer file, the card reader file, disk file, and the WORK2 disk area (cards 2 through 5).

The UCD mnemonics specify a card-to-disk operation (card 8). The default value of the *A* keyword parameter is assigned. The *B* keyword parameter specifies 90-byte output record and block lengths. A compare operation is specified by the *K2* keyword parameter.

The resulting printout shows all records unequal because the first two fields were rearranged and a 10-byte field was added to all disk file records during the copy operation.

5.4. Copy Disk-to-Disk Operation

In this example, the data stored in the WORK2 area of the disk file is copied to another area in the same disk file.

Job control stream

JOB CONTROL CARDS

```
          1        10        20        30        40        50
          |        |         |         |         |         |
 1.  // JOB DAMAIX,,A000
 2.  // DVC 20      // LFD PRNTR
 3.  // DVC 51      // VOL DSP028
 4.  // LBL WORK2   // LFD INPUT1
 5.  // DVC 51      // VOL DSP028
 6.  // EXT MI,C,,CYL,10
 7.  // LBL WORK3   // LFD OUTPUT1,,INIT
 8.  // EXEC DATA
 9.  /$
10.  UDD DP,MY,S2
11.  /*
12.  /&
13.  // FIN
```

Printer output

```
DONAHUE, MARY M          0000789123          255          10          030          1200000789123
GIAGLONE, JOHN K         0000776312          270          11          035          1250000776312
LANGINES, SAMANTHA R     0000617892          270          11          040          1300000617892
LOTIERZO, HELENE S       0000678978          310          13          050          1400000678978
MATHEWS, SAMUEL T        0000776321          425          15          071          1450000776321
MATTHEWS, JANE L         0000667843          270          11          045          1500000667843
NORDICK, JOHN M          0000667712          425          14          055          1550000667712
PARYIN, THOMAS G         0000673124          425          14          060          1650000673124
```

The printer file, the disk file, input WORK2 area, and output WORK3 area are assigned to the job step (cards 2 through 7). The output area, WORK3, is allocated by the // EXT statement (card 6).

A disk-to-disk operation is specified by the mnemonic UDD. The input record and block lengths are taken from the VTOC, and the output record and block lengths are assumed to be equal to the input lengths. We have used the *DP* keyword parameter to request that records copied to the output area, WORK3, be written to the printer file. By using the keyword parameters *MY* and *S2*, printer mismatches are ignored and the printer output is double-spaced. The printer output is printed in the *list* format by the default value of the *TL* keyword parameter.

The printer output shows that the complete WORK2 area has been copied to the WORK3 disk area.

5.5. Compare Disk-to-Disk Operation

This example shows the job control stream used to compare the data stored in disk WORK2 area to the data stored in disk WORK3 area. Any unequal records are listed on the printer output.

Job control stream

```
                    1        10        20        30        40        50
                    |        |         |         |         |         |
                    1. // JOB CATUDT,,A000
                    2. // DVC 20    // LFD PRNTR
JOB                 3. // DVC 51    // VOL DSP028
CONTROL             4. // LBL WORK2 // LFD INPUT1
CARDS               5. // DVC 51    // VOL DSP028
                    6. // LBL WORK3 // LFD INPUT2
                    7. // EXEC DATA
                    8. /$
Uio CARD            9. UDD K2
                   10. /*
                   11. /&
                   12. // FIN
```

The card reader file, printer file, and disk file are assigned in the job control stream (cards 2, 3, and 5). The WORK2 disk area is defined as INPUT1, and WORK3 disk area as INPUT2 (cards 4 and 6).

A disk-to-disk operation is specified by the UDD mnemonic. The same disk files are used for this compare operation as were used for the previous copy operation, so our *A* and *B* keyword parameter values remain the same, although the value for the *A* keyword parameter is taken from the VTOC for INPUT1 file and the value for the *B* keyword parameter is taken from the VTOC for INPUT2 file. *K2* keyword parameter specifies a compare operation.

The printer output shows that all records in disk area WORK2 compare equally to the records in disk area WORK3. Had there been unequal records, they would have been displayed as they were in 5.3.

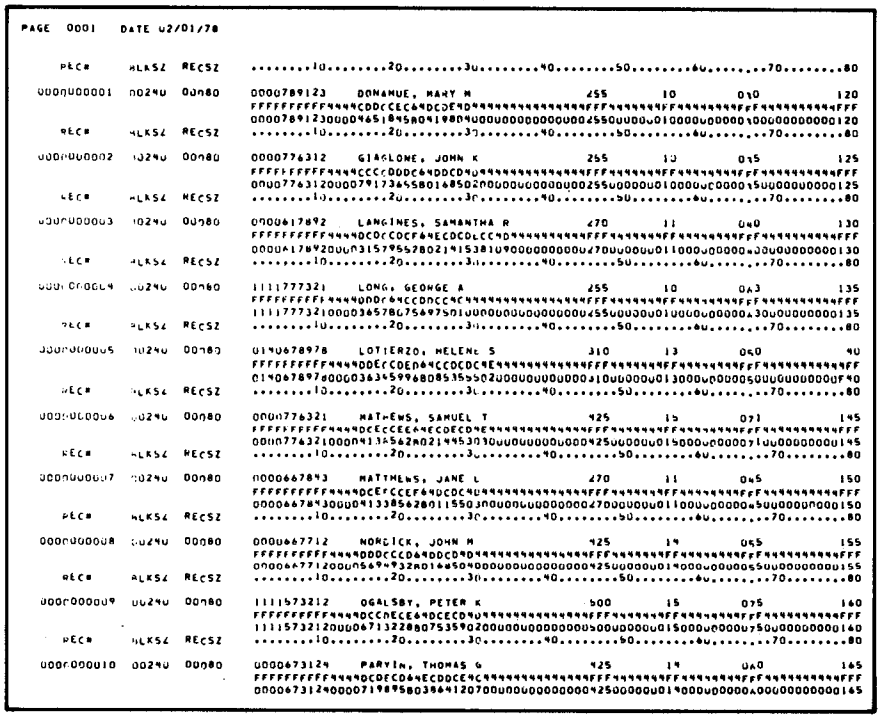
5.6. Copy Disk-to-Printer Operation

Shown in this example is the job control stream required to write the contents of the disk WORK2 area into the printer file.

Job control stream

```
                    1        10        20        30        40        50
                    |--------|---------|---------|---------|---------|
                 1. // JOB PRINTED,,A000
                 2. // DVC 51
JOB              3. // VOL DSP028
CONTROL          4. // LBL WORK2
CARDS            5. // LFD INPUT1
                 6. // DVC 20
                 7. // LFD PRNTR
                 8. // EXEC DATA
                 9. /$
Uio CARD        10. UDP MY,OB,PY,TD,S2
                11. /*
                12. /&
                13. // FIN
```

Printer output



The WORK2 area of the disk file is assigned as INPUT1 (cards 2 through 5). The mnemonic UDP signifies a disk-to-printer operation. A 90-byte record and block length input file is specified by values stored in the VTOC. The default values of the *B* keyword parameter, 90-byte record length, and 120-byte print line, are acceptable for this job step. Printer mismatches are ignored by use of the *MY* keyword parameter. The output is printed in hexadecimal and EBCDIC as specified by the keyword parameter *OB*. Page numbers and dates are printed by specifying the *PY* keyword parameter. The *TD* keyword parameter specifies printing the output in the display format.

5.7. Correction Operation

In this example, we will correct one of the records on the WORK3 area of the disk file by using the COR statement.

The amount of the hourly rate and employee class rating for John K. Giaglone will be changed from 255 to 270 for the hourly rate, and from 10 to 11 for the employee class. All other information in this record remains the same.

Job control stream

```
                    1        10        20        30        40        50        60        70        80
                    |--------|---------|---------|---------|---------|---------|---------|---------|
                 1. // JOB CORRECT,,A000
                 2. // DVC 20        // LFD PRNTR
                 3. // DVC 51        // VOL DSP028
JOB              4. // LBL WORK3     // LFD INPUT1
CONTROL          5. // DVC 51        // VOL DSP028
CARDS            6. // EXT MI,C,,CYL,10
                 7. // LBL WORK4     //LFD OUTPUT1,,INIT
                 8. // EXEC DATA
                 9. /$
Uio CARD        10.   UDD B=(90,512),MY
UTILITY         11.   COR N=(1),A=(2),B=(2)
MODIFIER        12.  00090GIAGLONE,  JOHN K          000077 6312          270        11          035
CARDS           13.           125 0000776312
                14. /*
                15. /&
                16. // FIN
```

Printer output

DONAHUE, MARY M	0000789123	255	10	030	1200000789123
GIAGLONE, JOHN K	0000776312	270	11	035	1250000776312
LANGINES, SAMANTHA R	0000617892	270	11	040	1300000617892
LOTIERZO, HELENE S	0000678978	310	13	050	1400000678978
MATHEWS, SAMUEL T	0000776321	425	15	071	1450000776321
MATTHEWS, JANE L	0000667843	270	11	045	1500000667843
NORDICK, JOHN M	0000667712	425	14	055	1550000667712
PARYIN, THOMAS G	0000673124	425	14	060	1650000673124

Device assignment, allocation of main storage and disk space, and initiation of the DATA routine are included in the job control cards (cards 1 through 9). Note that card 6 is included to allocate disk space for the WORK4 area. This new area on the disk file is required because our original area (WORK3) cannot be corrected, but must be written to another area while corrections are being made.

The UDD mnemonic specifies a disk-to-disk operation. The input block and record lengths are taken from the VTOC. The *B* keyword parameter specifies that the output record length is 90 bytes and the block length is also 90. The records are fixed length. The *MY* keyword parameter specifies that printer mismatches are to be ignored.

In the COR statement, the keyword parameter *N* specifies that one record follows. The keyword parameter *A* specifies that the starting location of the record to be corrected is record 2 and the keyword parameter *B* specifies the ending of the correction (card 11). Lines 12 and 13 specify the correction that is to be made to record 2.

The printer output shows that the corrections to this record have been executed successfully.

Section 6
Job Control Procedures (Jprocs)

6.1. Purpose and Application

A job control procedure (jproc) is a series of job control statements that establishes a sequence of steps that lead to the solution of a problem; that is, the proper control stream needed to assign the files and devices used by your job. This jproc is used to eliminate the need to repeatedly code identical sequences of job control statements. The Unisys Operating System/3 (OS/3) allows you to code a single statement (a jproc call statement) that generates the proper job control statement sequence for you.

Unisys has supplied a set of jprocs that contain the job control statements that are needed to execute the data utilities. These jprocs are stored in the job control stream library file (YJCS). In order to use them, you place a jproc call statement in your control stream at the point where you would have otherwise placed the job control statements (if you had used them). The jproc call statement then generates the proper sequence of job control statements depending on values you supply through positional and keyword parameters. Refer to the *Job Control Programming Guide* (UP-9986), for detailed information on writing and calling procedure definitions.

The data utility jprocs can be used to simplify the job control stream when you run a disk-to-disk (UDD), disk-to-tape (UDT), or tape-to-disk (UTD) operation. The jproc call statement can also be used when you compare two files. This method can be used to shorten the job control stream described in 4.2.12. However, both methods are functionally identical and use of either method is at your discretion.

6.1.1. Combination of File Types

Tape or disk files may be copied or relocated to the same volume or to a different volume by use of a jproc call statement. For disk input files, the output files can be either a fixed or variable disk, or a fixed or variable tape.

6.1.2. Job Control Requirements

Your operations with job control are simplified with a jproc call statement. The UDD, UDT, and UTD statements generate the EXEC job control statement and all other device assignment cards you need to run a disk-to-disk, disk-to-tape, or tape-to-disk operation. To ensure compatibility between disk and tape unit numbers, the DVCVOL jproc call statement assigns logical unit numbers for disk units, and the DVCVTP jproc call statement assigns logical unit numbers for tape units. Refer to the *Job Control Programming Guide* (UP-9986).

The basic structure of your job control stream when using a jproc call statement is as follows:

- JOB control statement (// JOB parameters)
- Data utility jproc call statement (// UDD or // UDT or // UTD)
- Start-of-data job control statement (/$)
- Data utility control statement (Refer to 4.2.5.)
- End-of-data job control statement (/*)
- End-of-job control statement (/&)
- FIN job control statement (// FIN)

6.2. Job Control Procedures

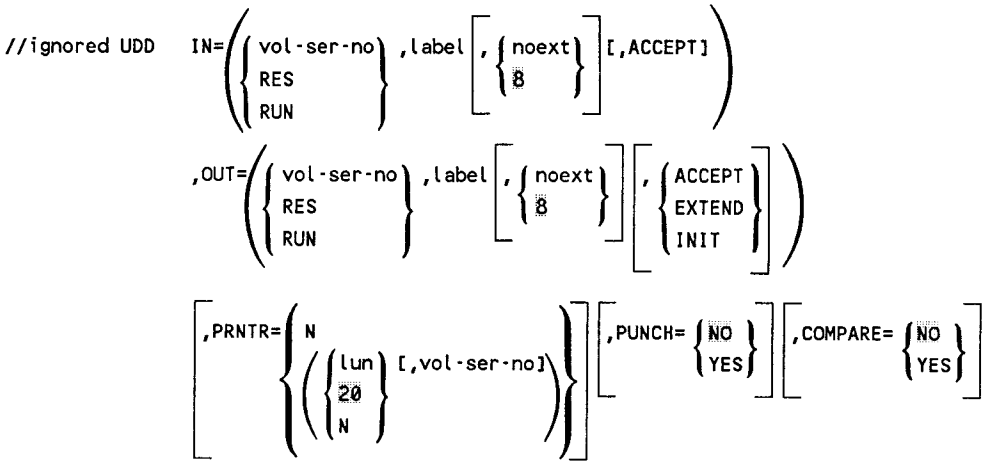
The three types of job control procedures (UDD, UDT, and UTD) are described in 6.2.1 through 6.2.3.

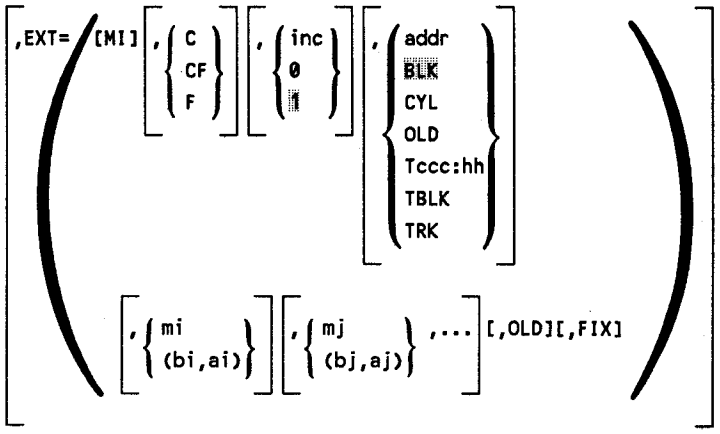
6.2.1. UDD Job Control Procedure

Function

UDD generates the job control statements for your device assignment sets that are required by the DATA routine to copy or compare one disk file to another disk file (disk-to-disk operation).

Format





Label

The label field is ignored.

Parameters

IN=

Supplies the information required to define the input file. The information is coded within the parentheses.

where:

vol-ser-no

Specifies the volume serial number of the volume containing the input file.

RES

Specifies that the input file is located on the SYSRES.

RUN

Specifies that the input file is located on the volume containing the job's YRUN file.

label

Specifies the file identifier for the input file.

noext

Specifies the number of extents in the file requiring space to be reserved in the prologue. The default value is 8.

ACCEPT

Indicates that the data management specifications should be obtained from format 1 and format 2 labels in the VTOC. This specification is used for files previously opened and closed by data management. Normally omitted since data utilities performs the operation.

OUT=

Supplies information required to define the output file for a copy operation or to define a secondary input file for a compare operation.

where:

vol-ser-no

Specifies the volume serial number of the disk containing the output (or secondary input) file.

RES

Specifies that the output (or secondary input) file is located on the SYSRES disk.

RUN

Specifies that the output (or secondary input) file is located on the volume containing the job's YRUN file.

label

Specifies the file identifier for the output (or secondary input) file.

noext

Specifies the number of extents in the file requiring space to be reserved in the job prologue. The default value is 8.

ACCEPT

Indicates that the data management specifications should be obtained from format 1 and format 2 labels in the VTOC. This specification is used for files previously opened and closed by data management. Normally omitted, since data utilities performs the operation.

EXTEND

Indicates that the MIRAM (MI) file is to be extended. The information is appended to the present end of the file.

INIT

Indicates that the specified file is to be initialized starting with the first record each time the file is opened. Previous control information in the format labels is ignored at the file open time, and the information is overwritten by specifications contained in this DVC-LFD sequence at file close time.

PRNTR=

Indicates whether a device assignment set is to be generated for the printer and defines the print file.

where:

N
: Suppresses the generation of a device assignment set for the printer. The device assignment set is to be manually inserted. This allows for the insertion of SPL, LCB, and VFB job control statements.

{ lun / 20 }
: Specifies the logical unit number for the printer. The default value is 20.

vol-ser-no
: Provides a 1- to 6-alphanumeric-character remote destination identifier for the print file when dealing with remote job entry.

PUNCH=YES
: Generates the device assignment set for the card punch. This parameter is required when the punch dual output feature (DC) is selected on the data control card.

PUNCH=NO
: No device assignment set is generated for the card punch. This is the default value and need not be specified.

COMPARE=YES
: Specifies that the compare operation (K2) is selected on the data control card. This keyword causes the file name INPUT2 (secondary input) to be generated for the file specified by the *OUT* keyword parameter.

COMPARE=NO
: Specifies that this is a copy operation. This is the default value and need not be specified.

EXT=
: Specifies the extent specifications to be used when reserving system resources for an output file. The information is supplied as a subparameter list enclosed within parentheses.

where:

MI
: Indicates this is a MIRAM file.

C
: Indicates that the file must be allocated contiguously.

F
: Indicates that the file is to be formatted at allocation time. You must select *BLK* for Subparameter 4.

CF

Indicates that both of the preceding parameters apply for Subparameter 2.

Note: *If Subparameter 2 is omitted (C, F, or CF), none of the above options apply.*

inc

Specifies the secondary increment (in cylinders) by which the file is to be extended if automatic extension is required.

0

Specifies there can be no dynamic extension of the file.

1

Indicates one cylinder block increment is specified.

Note: *If there is no EXT keyword parameter for this file, the value of the most recently specified secondary increment is implemented.*

addr

Specifies the absolute cylinder address where the file is to begin. Allocation is specified in terms of cylinders.

BLK

Indicates that allocation is in terms of blocks. This is the default value if no allocation type is specified.

CYL

Indicates that allocation is in terms of cylinders.

OLD

Indicates that the secondary allocation increment (subparameter 3) is changed. No additional space is allocated by job control. If specified, it must be the last subparameter in the list.

Tccc:hh

Specifies the absolute track address (hexadecimal) in cylinder/head format where the file is to begin. Allocation is in terms of tracks.

TBLK

Allocates space in blocks; actual allocation is in terms of tracks.

TRK

Allocates space in tracks.

mi and mj

Specifies the number of cylinders to allocate for this file. Subparameter 4 must be *CYL* or *addr*.

(bi,ai) and (bj,aj)

This option is used when allocation is in terms of blocks, where *bi* is the average block length, and *ai* is the number of blocks to allocate.

Subparameters 6 through n:

These subparameters are essentially the same as Subparameter 5 and are used to describe additional extents, where n $\leq$ 20. This allows you to specify up to 16 extents.

Subparameter n + 1:

OLD

Indicates that this extent applies to a previously allocated file. Primary members specified by this keyword parameter will be allocated by job control and added to the existing file. If this parameter is omitted, the request is for a new extent.

Subparameter n + 2:

FIX

Indicates this extent is in a fixed-head area of an 8417 disk.

Example 1a

This example illustrates a disk-to-disk copy operation using the UDD jproc call statement for a simple disk-to-disk copy. It is assumed that space has been previously allocated for the output file. In this example, the input and output files are on the same disk.

```
   1        10        20        30        40        50        60
1. // JOB DCDCCPY1,,A000
2. // UDD IN=(DSP001,LABEL1),OUT=(DSP001,LABEL2,,INIT)
3. /$
4. UDD
5. /*
6. /&
7. // FIN
```

Line	Explanation
1	Indicates the name of the job is DCDCCPY1 and specifies the amount of main storage in hexadecimal (A000).

Line	Explanation
2	Indicates the name of the jproc being called is UDD. The *IN* keyword parameter indicates the file identifier (LABEL1) of the input file and the file and serial number (DSP001) of the disk where the file is located. The *OUT* keyword parameter indicates the file identifier (LABEL2) of the output file and the volume serial number (DSP001) of the disk where the file is located. Since the file is assumed to be previously allocated, *INIT* is used to indicate that any information that may be on the file is to be overwritten by the new information.
3	Indicates the start of data
4	Indicates the data utility control statement. The input and output file type are both defaulted to the input file type specified in the VTOC, and the default block size and record size for input and output are 80.
5	Indicates the end of data
6	Indicates the end of the job
7	Indicates termination of card reader operation

Example 1b

This example illustrates the expanded job stream generated by Example 1a.

```
   1        10        20        30
   |--------|---------|---------|
1. | // JOB DCDCCPY1,,A000
2A.| // DVC 20
2B.| // LFD PRNTR
2C.| // DVC 50
2D.| // VOL DSP001
2E.| // LBL LABEL1
2F.| // LFD INPUT1
2G.| // DVC 50
2H.| // VOL DSP001
2I.| // LBL LABEL2
2J.| // LFD OUTPUT1,,INIT
2K.| // EXEC DATA
3. | /$
4. | UDD
5. | /*
6. | /&
7. | // FIN
```

Line	Explanation
1	Same as for Example 1a
2A-2B	These lines assign the printer to the job. The default logical unit number is 20 for this example.
2C-2F	Indicates that the input file has a file identifier of LABEL1 and a file name of INPUT1 and is located on the disk with volume serial number DSP001. This information is obtained from the *IN* keyword parameter in line 2 of Example 1a. Volume serial number DSP001 is assigned to logical unit number 50.
2G-2J	Indicates that the output file has a file identifier of LABEL2 and a file name of OUTPUT1 and is on the same volume as the input file (DSP001). The same logical unit number is assigned (50). This information is obtained from the *OUT* parameter in line 2 of Example 1a.
2K	This line loads the data utility program (DATA) from YLOD.
3-7	Same as for Example 1a

Example 2a

This example illustrates a MIRAM-to-MIRAM disk copy using the UDD procedure call statement. Space for the output file is allocated to the file in this job (but before execution). The input file is printed in single-space display format as it is being copied.

```
    1        10        20        30        40        50        60        72
1.  // JOB DCDCCPY2,,A000
2.  // UDD IN=(DSP001,LABEL1),                                          X
3.  //1 OUT=(DSP002,LABEL2),                                           X
4.  //2 EXT=(MI,C,,CYL,3)
5.  /$
6.  UDD  DP
7.  /*
8.  /&
9.  // FIN
```

Line	Explanation
1	Indicates the name of the job is DCDCCPY2.
2	Indicates the name of the jproc being called is UDD. The *IN* keyword parameter indicates the file identifier (LABEL1) of the input file and the volume serial number (DSP001) of the disk where the file is located.

Line	Explanation
3	Indicates a continuation of the UDD jproc call. The *OUT* keyword parameter indicates the file label (LABEL2) of the output file and the volume serial number (DSP002) of the disk where the file is located.
4	Indicates further continuation of the UDD procedure call; space for the file is to be allocated on the output disk. This line supplies information required to allocate the file.
5	Indicates the start of data
6	Data utility statement; the disk input file characteristics are taken from the VTOC. The *DP* keyword indicates the input file is to be printed in addition to the output file. The default block and record sizes for input and output are 80.
7	Indicates the end of data
8	Indicates the end of job
9	Indicates the termination of card reader operation

Example 2b

This example illustrates the expanded job stream generated by Example 2a.

```
   1        10        20        30        40
   |--------|---------|---------|---------|
1. // JOB DCDCCPY2,,A000
2A. // DVC 20
2B. // LFD PRNTR
2C. // DVC 50
2D. // VOL DSP001
2E. // LBL LABEL1
2F. // LFD INPUT1
2G. // DVC 51
2H. // VOL DSP002
2I. // EXT MI,C,,CYL,3
2J. // LBL LABEL2
2K. // LFD OUTPUT1
2L. // EXEC DATA
5.  /$
6.  UDD  DP
7.  /*
8.  /&
9.  // FIN
```

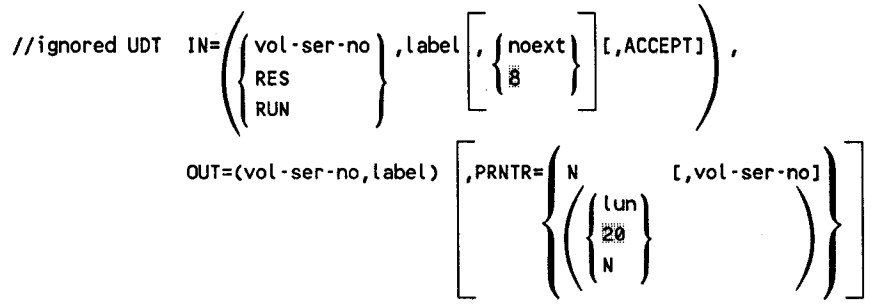
Line	Explanation
1	Same as for Example 2a
2A-2B	These lines assign the printer to the job. The default logical unit number is 20 for this example.
2C-2F	Indicates that the input file has a file identifier of LABEL1 and a file name of INPUT1 and is on the disk with volume serial number DSP001. This information was obtained from the *IN* keyword in line 3 of Example 2a. The volume is assigned to logical unit number 50.
2G, 2H, 2J	Indicates that the output file has a file identifier of LABEL2 and a disk volume serial number of DSP002. This information was obtained from the *OUT* keyword parameter in line 3 of Example 2a. The volume is assigned to logical unit number 51.
2I	Indicates that space is to be allocated for a MIRAM file on three contiguous cylinders. This is obtained from line 4 of Example 2a. By default, one cylinder is provided for dynamic extension.
2K	Indicates the file name for the output file is OUTPUT1
2L	Indicates the DATA routine is loaded from YLOD
5-9	Same as for Example 2a

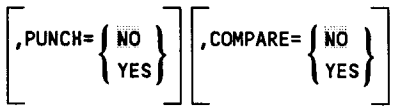
6.2.2. UDT Job Control Procedure

Function

UDT generates the job control statements for your device assignment sets that are required by the data utility routine to copy or compare a disk file to a tape file (disk-to-tape operation).

Format





Label

The label field is ignored.

Parameters

IN=

Supplies the information required to define the input file. This information is coded within the parentheses.

where:

vol-ser-no

Specifies the volume serial number of the disk volume containing the input file.

RES

Specifies that the input file is located on the SYSRES disk.

RUN

Specifies that the input file is on the volume containing the job's YRUN file.

label

Specifies the file identifier for the input file.

noext

Specifies the number of extents in the file for which space must be reserved in the prologue.

ACCEPT

Indicates that the data management specifications should be obtained from the format 1 and format 2 labels on the VTOC. This specification is used for files previously opened and closed by data management. It is normally omitted, since data utilities performs the operation.

OUT=

Supplies information required to define the output file for a copy operation or a secondary input file for a compare operation. The information is coded within parentheses.

where:

vol-ser-no
: Indicates the volume serial number of the disk containing the output (or secondary input) file.

label
: Specifies the file identifier for the output (or secondary input) file.

PRNTR=
: Indicates whether a device assignment set is to be generated for the printer and defines the print file.

where:

N
: Suppresses the generation of a device assignment set for the printer. The device assignment set is to be manually inserted. This allows for the insertion of LCB and VFB job control statements.

{ lun / 20 }
: Specifies the logical unit number for the printer. The default value is 20.

vol-ser-no
: Provides a 1- to 6-alphanumeric-character remote destination identifier for the print file when dealing with remote job entry.

PUNCH=YES
: Generates a device assignment set for the punch. This function is required when the card punch dual output feature (DC) is selected on the data control card.

PUNCH=NO
: No device assignment set is generated for the punch. This is the default value and need not be specified.

COMPARE=YES
: Specifies that compare operation option K2 will be selected on the data control card. This keyword causes the file name INPUT2 (secondary input) to be generated for the file specified by the *OUT* keyword parameter.

COMPARE=NO
: Specifies that this is a copy operation. This is the default value and need not be specified.

Example 1a

This example illustrates a disk-to-tape copy operation using the UDT jproc call statement for copying a disk to a standard labeled tape. The first 80 characters of each record are to be punched.

```
    1        10        20        30        40        50        60
    |--------|---------|---------|---------|---------|---------|
1.  // JOB DCTP01,,A000
2.  // UDT IN=(DSP001,LABEL1),OUT=(SP0001,LABEL2),PUNCH=YES
3.  /$
4.  UDT DC,OR,LO
5.  /*
6.  /&
7.  // FIN
```

Line	Explanation
1	Indicates the name of the job is DCTP01.
2	Indicates the name of the jproc being called is UDT. The *IN* parameter indicates the file identifier is (LABEL1) of the input file and the volume serial number (DSP001) of the disk where the file is located. The *OUT* keyword parameter indicates the tape file identifier is LABEL2 and the tape volume serial number is SP0001. The *PUNCH* parameter indicates that the *DC* keyword parameter (Table 3-3) will be used on the data utility control statement; therefore the output will also be punched.
3	Indicates the start of data
4	This is a data utility control statement indicating the disk input and tape output parameters. The *DC* parameter indicates punched output in addition to tape output. The *OR* parameter indicates the tape is to be rewound before and after processing. The *LO* parameter indicates standard tape labels are used. the default block size and record size for input and output are 80.
5	Indicates the end of data
6	Indicates the end of the job
7	Terminates the card reader operation

Example 1b

This example illustrates an expanded job stream generated by Example 1a.

```
    1        10        20        30        40
    |--------|---------|---------|---------|
1.  // JOB DCTP01,,A000
2A. // DVC 20
2B. // LFD PRNTR
2C. // DVC 50
2D. // VOL DSP001
2E. // LBL LABEL1
2F. // LFD INPUT1
2G. // DVC 90
2H. // VOL SP0001
2I. // LBL LABEL2
2J. // LFD OUTPUT1
2K. // DVC 40
2L. // LFD OUTPUT2
2M. // EXEC DATA
3.  /$
4.  UDT DC,OR,LO
5.  /*
6.  /&
7.  // FIN
```

Line	Explanation
1	Same as Example 1a
2A-2B	Assigns the printer to this job. The default logical unit number 20 is used.
2C-2F	Indicates the input file has a file identifier of LABEL1 and a file name of INPUT1 and is located on the disk having volume serial number DSP001. The volume (DSP001) is assigned to logical unit number 50. This information was obtained from the *IN* keyword in line 2 of Example 1a.
2G-2J	Indicates the output file has a file identifier of LABEL2, a file name of OUTPUT1, and a tape volume serial number of SP0001. The volume is assigned to logical unit number 90. This information was obtained from the *OUT* keyword in line 2 of Example 1a.
2K-2L	Assigns the card punch (logical unit number 40) to the job. The file name is OUTPUT2.

Line	Explanation
2M	Loads the DATA routine from YLOD
3-7	Same as Example 1a

6.2.3. UTD Job Control Procedure

Function

UTD generates the job control statements for your device assignment sets that are required by the data utility routine to copy or compare a tape file to a disk file (tape-to-disk operation).

Format

```
//ignored UTD IN=(vol-ser-no,label)
,OUT=( {vol-ser-no} ,label [,{noext}] [,{ACCEPT}] )
       {RES       }         {8    }   {EXTEND }
       {RUN       }                   {INIT   }
[,PRNTR={N                                    }] [,PUNCH={NO }] [,COMPARE={NO }]
        {({lun} [,vol-ser-no])                }         {YES}          {YES}
        {({20 }              )                }
        {({N  }              )                }
[,EXT=( [MI] [,{C }] [,{inc }] [,{addr  }]                    )]
             {CF}    {0   }    {BLK   }
             {F }    {1   }    {CYL   }
                               {OLD   }
                               {Tccc:hh}
                               {TBLK  }
                               {TRK   }
        [,{mi   }] [,{mj   }] ,... [,OLD][,FIX]
          {(bi,ai)}  {(bj,aj)}
```

Label

The label field is ignored.

Parameters

IN=
Supplies the information required to define the input file. This information is coded within the parentheses.

where:

vol-ser-no
Specifies the volume serial number of the volume containing the input file.

label
Specifies the file identifier for the input file.

OUT=
Supplies the required information to define the output file for a copy operation or secondary input for a compare operation.

where:

vol-ser-no
Provides the volume serial number of the disk containing the output file (or secondary input).

RES
Indicates the output (secondary input) file is located on the SYSRES disk.

RUN
Indicates the output (secondary input) file is located on the volume containing the job's YRUN file.

label
Specifies the file identifier for the output (or secondary input) file.

noext
Specifies the number of extents in the file to be reserved in the extent table storage for use by the data access method. The default value is 8.

ACCEPT
Indicates that the data management specification should be obtained from format 1 and format 2 labels in the VTOC. This specification is used for files previously opened and closed by data management. Normally omitted, since data utilities performs the operation.

EXTEND
Indicates that a MIRAM file is to be extended. The information is appended to the present end of the file.

INIT

Indicates that the specified file is to initialize starting with the first record each time the file is opened. Previous control information in the format labels is ignored at file open time and is overwritten by specifications contained in this DVC-LFD sequence at file closure time.

PRNTR=

Indicates whether a device assignment set is to be generated for the printer and defines the printer file.

where:

N

Suppresses the generation of a device assignment set for the printer. The device assignment set is to be manually inserted. This allows for the insertion of SPL, LCB, and VFB job control statements.

{ lun / 20 }

Specifies the logical unit number for the printer. The default value is 20.

vol-ser-no

Provides a 1- to 6-alphanumeric-character remote destination identifier for the print file when dealing with remote job entry.

PUNCH=YES

Generates a device assignment set for the punch. This is required when the punch dual output feature (DC) is selected on the data utility control card.

PUNCH=NO

No device assignment set is generated for the punch. This is the default value and need not be specified.

COMPARE=YES

Indicates that compare operation option K2 is selected on the data utility card. This keyword generates a file name of INPUT2 (secondary input) for the file specified by the *OUT* parameter.

COMPARE=NO

Specifies that this is a copy operation. This is the default value and need not be specified.

EXT=

Supplies the extent specifications to be used when reserving system resources for an output file. The information is supplied as a subparameter list enclosed in parentheses.

where:

MI
: Indicates this is a MIRAM file.

C
: Indicates the file must be allocated contiguously.

F
: Indicates the file is to be formatted at allocation time. Subparameter 4 must be *BLK*.

CF
: Indicates both of the preceding parameters apply to Subparameter 2.

Note: *If this parameter is omitted (C, F, or CF), none of the above options apply.*

inc
: Specifies the secondary increment (in cylinders) by which the file is to be extended if automatic extension is required.

0
: Indicates that there can be no dynamic extension of the file.

1
: Indicates an extension of one cylinder increment of the file.

Note: *If there is no EXT keyword parameter for this file, the value of the most recently specified secondary increment is used.*

addr
: Specifies the absolute cylinder address where the file is to begin. The allocation is in terms of cylinders.

BLK
: Indicates that the allocation is in terms of blocks. This is the default parameter.

CYL
: Indicates that the allocation is in terms of cylinders.

OLD
: Indicates that the secondary allocation increment is changed (subparameter 3). No additional space is allocated by job control. If specified, this must be the last subparameter in the sublist.

Tccc:hh
: Specifies the absolute track address (hexadecimal) in cylinder/head format where the file is to begin. Allocation is in terms of tracks.

`TBLK`
Allocates space in blocks; actual allocation is in terms of tracks.

`TRK`
Allocates space in tracks.

`mi and mj`
Specifies the number of cylinders to allocate for this file. Subparameter 4 must be *CYL* or *addr*.

`(bi,ai) and (bj,aj)`
This option is used when allocation is in terms of blocks, where *bi* is the average block length, and *ai* is the number of blocks to allocate.

`(pi %,ci)`
This is for split cylinder allocation only.

- This parameter specifies two conditions: the number of tracks allocated as a percentage of the total number of tracks per cylinder.
- The percent sign (%) must be coded as shown. This option is not used with regular cylinder allocation.

Subparameters 6 through n:

Specified in the same format as Subparameter 5 and used to describe additional extents, where n $\leq$ 20. This allows you to specify up to 16 extents.

Subparameter n + 1:

`OLD`
Indicates this extent applies to a previously allocated file. Primary members specified by this keyword parameter are allocated by job control and added to the existing file. If this parameter is omitted, the request is for a new extent.

Subparameter n + 2:

`FIX`
Indicates this extent is in a fixed-head area of an 8417 disk.

Example 1a

This example illustrates a tape-to-disk copy using the UTD procedure call statement. The operation uses variable-length records. It assumes that the disk area has been previously allocated.

```
    1        10        20        30        40        50        60
1.  // JOB TPDC01,,A000
2.  // UTD IN=(SP001,LABEL1),OUT=(DSP001,LABEL2,,INIT)
3.  /$
4.  UTD A=(5,89),FV,LO
5.  /*
6.  /&
7.  // FIN
```

Line	Explanation
1	Indicates the name of the job is TPDC01.
2	Indicates the name of the jproc being called is UTD. The *IN* parameter indicates the tape file identifier is LABEL1 and the tape volume serial number is SP001. The *OUT* keyword parameter indicates that the disk file identifier is LABEL2 and the volume serial number is DSP001. Since the file is assumed to be previously allocated, *INIT* is used to indicate that any information that may be on the file is to be overwritten by the new information.
3	Indicates the start of data
4	Specifies a data utility control statement indicating tape input and disk output. The *FV* parameter indicates that records are of variable length. The *A* parameter indicates the maximum block size is 89, and the minimum record size is 5. With the *B* parameter not specified, maximum block size defaults to the maximum block size for input. The *LO* parameter indicates the tape has standard labels.
5	Indicates the end of data
6	Indicates the end of the job
7	Terminates the card reader operation

Example 1b

This example illustrates an expanded job stream generated by Example 1a.

```
   1        10        20        30        40
   |--------|---------|---------|---------|
1. // JOB TPDC01,,A000
2A. // DVC 20
2B. // LFD PRNTR
2C. // DVC 90
2D. // VOL SP001
2E. // LBL LABEL1
2F. // LFD INPUT1
2G. // DVC 50
2H. // VOL DSP001
2I. // LBL LABEL2
2J. // LFD OUTPUT1,,INIT
2K. // EXEC DATA
3. /$
4. UTD A=(5,89),FV,LO
5. /*
6. /&
7. // FIN
```

Line	Explanation
1	Same as for Example 1a
2A-2B	Assigns the printer to the job. The default logical unit number 20 is used.
2C-2F	Indicates the input file has a file identifier of LABEL1 and a file name of INPUT1 and the tape volume serial number is SP001. It is assigned logical unit number 90. This information is obtained from the *IN* keyword parameter in line 2 of Example 1a.
2G-2J	Indicates the output file has a file identifier of LABEL2, a file name of OUTPUT1, and a volume serial number of DSP001. It is assigned logical unit number 50. This information is obtained from the *OUT* parameter in line 2 of Example 1a.
2K	Loads the DATA routine from YLOD
3-7	Same as for Example 1a

Section 7
MILOAD Utility - Special Purpose Loader For MIRAM Files

7.1. MILOAD - Why You Need It

Loading MIRAM files by using conventional loading techniques can result in long load times due to the number of I/O operations required to create the dynamic indexes for these files. You can significantly increase performance by using MILOAD, the more efficient fast-loader utility program, whenever you need to load a MIRAM file. There are, however, some restrictions and trade-offs to consider before using this utility.

7.1.1. Considerations before Using MILOAD

- User Responsibilities for Input Files

  As a user, you are responsible for preparing the input files processed by MILOAD. You must make certain that these files consist only of records that can be successfully added to the MIRAM file being created. For example, if you instruct MILOAD not to accept records that contain duplicate keys, then the input file to MILOAD must not contain records having duplicate keys. In addition, records containing illegal duplicate keys must be removed (sorted) from the input to MILOAD. To do this, use the standard OS/3 SORT program. You are also responsible for defining the criteria for record selection. Your input files can reside on disk or tape and tape files must be created by OS/3 data management in EBCDIC mode.

- Output File Structure

  The output file created by MILOAD must be a MIRAM characteristic file. You can force all keyed MIRAM files to be MIRAM characteristic via the SUPGEN MIRAMCHAR=YES keyword parameter. If you do not use this parameter, the file must include one or more of the following criteria:

  - More than one key
  - Duplicate keys
  - Keys that may be changed during update operations
  - A variable file record format
  - A record control byte (RCB)

- 8430/33 Disks

  When you create the output file for MILOAD on 8430/33 disks, we recommend that you avoid having data management automatically compute the physical sector size (PSS) for data partitions (VSEC=YES). The reason for this recommendation is that MILOAD uses large buffers to achieve performance; by specifying VSEC=YES, you create a situation where every program that later accesses the output file must contend with the inconvenience of large buffers. Therefore, performance is degraded. You may, however, specify a value for the PSS (VSEC=n), but make certain that you choose a value that is comfortable to work with throughout the life of the file. For additional information about VSEC, see the current version of the *Consolidated Data Management Programming Guide,* (UP-9978).

- Need for Temporary Work Files

  MILOAD requires temporary disk work files to perform the sort process, which is an integral part of its operation. The amount of work file space required depends on the number of records involved and the key characteristics of the output file. The algorithm for calculating the amount of work file space needed for a given job is found in a later part of this section.

- Reading INPUT1 File by KEY

  MILOAD supports the reading of the INPUT1 file by key. This means that the data in the OUTPUT1 file can be organized in any key sequence simply by reading the indexed INPUT1 file by that key of reference and generating a new OUTPUT1 file. This is useful when there is a requirement for unkeyed sequential processing in a specific key's sequence.

- Rebuilding an Existing File's Index in Place

  MILOAD allows you to rebuild an existing file's index in place. Only one device assignment (OUTPUT1) should be specified. The data is left intact and the index is rebuilt. This is particularly useful when a MIRAM file has to be reloaded because of a compromised index condition. There is a performance gain when you rebuild your file with this option. There is also a reduction in the disk space required because there is no need for an INPUT1 file.

- Duplicate Key Handling

  The primary objective of MILOAD is to increase performance in loading MIRAM files. Achieving this goal affects flexibility, temporary disk file space, and main storage requirements. For example, MILOAD does not provide for record selection criteria and terminates abnormally for any hardware or logical error condition. Data-related errors such as illegal duplicate keys are recoverable when using MILOAD. The INPUT1 file can contain records with duplicate keys for those OUTPUT1 key structures that do not permit duplicates.

  When an illegal duplicate is detected, MILOAD issues a warning message on the

When an illegal duplicate is detected, MILOAD issues a warning message on the screen. The key of reference, the relative record number, and the illegal key value (which may or may not be printable) are displayed on the printer listing. This key is not loaded into the index; however, the record still exists in the data partition of the OUTPUT1 file. Processing continues to completion. The OUTPUT1 file is accessible but incomplete. You can retrieve the illegal duplicate record unkeyed but you cannot read it keyed. You should do some additional clean-up work on OUTPUT1 to resolve the illegal duplicate record problem. This can be done by using MILOAD again and, at this time, retrieving records by a key that does not allow duplicates (see the MKR parameter).

- Hardware Requirements

  MILOAD supports the following disk devices: 8417, 8418, 8419, 8430, 8433, 8470, 8494, 9493, and M9720.

7.1.2. Restrictions

The output file is not usable until MILOAD successfully completes. Logical errors, hardware errors, and asynchronous termination leave the output file that is created by MILOAD in a compromised state. It is your responsibility to ensure the successful termination of MILOAD.

MILOAD does not support the select statement (SEL) on copy operations.

Sharing the output file with other jobs is not permitted. Use of the output file must be exclusive until MILOAD successfully loads all the records of a file.

7.1.3. Trade-Offs

MILOAD requires temporary disk file space for use by the SORT routine. This temporary disk file space could become a significant amount if the ratio of key data to record length is high. It could approach the size of the file being loaded.

MILOAD requires 65K of main storage plus an additional amount for the sort work space and buffers. The total amount of main storage needed to execute MILOAD must be 85K bytes or greater.

7.2. Using MILOAD for Creating MIRAM Characteristic Files

After reading 7.1, if you determine that you have the necessary resources and main storage to use MILOAD, you can easily put it to work loading your MIRAM files. Your interface with MILOAD is through the OS/3 job control language (JCL).

The JCL assignment sets for your input, output, scratch, and print file devices are consistent with those used for running any job in an OS/3 environment. The control statements that define your file characteristics and execute MILOAD are compatible with the input and output statements used by the DATA routine. (See "Uio Parameters" in Section 4 and Table 4-1.) Examples are provided to assist you in preparing device assignment sets and run control statements.

7.2.1. Device Assignment Sets for Defining Your Files

Your MILOAD job stream must include device assignment sets for all your input, output, scratch, and print files.

- Input File

  The input file to MILOAD, as previously mentioned, can reside on disk or tape. You are required to identify the device type, its volume serial number, and the name of the file containing the input records for processing; you must also define the logical file name as INPUT1. The following examples illustrate the device assignment sets for disk and tape input files of your job stream:

  Disk Input

  ```
  // DVC 50
  // VOL vsn
  // LBL filename
  // LFD INPUT1
  ```

  Tape Input

  ```
  // DVC 90
  // VOL vsn
  // LBL filename
  // LFD INPUT1
  ```

- Output File

  The output file created by MILOAD can only be a MIRAM characteristic file and must reside on disk. As with the input file, you must identify the device type to which the output is written, its volume serial number, and the name of the file to contain the output records. You must also specify the logical name of the file as OUTPUT1. The device assignment set for your output file is:

  ```
  // DVC 50
  // VOL vsn
  // EXT MI,,,CYL,mi
  // LBL filename
  // LFD OUTPUT1,,INIT
  ```

  In this example, the INIT parameter indicates that the output file is an existing file that is reinitialized. You could omit the INIT parameter from the LFD statement for a new file. The EXT statement is required if file space has not been previously allocated.

  If you want the output file created with recovery included, add a // DD RECV=YES to your assignment set.

- Scratch File

MILOAD requires temporary work file space on disk for use by the SORT routine. The device assignment set for the scratch file not only identifies the device type, vsn, filename, and logical description; it also allocates the amount of temporary space required. This is a typical example of the device assignment set to be included in your job stream for a scratch file:

```
// DVC 50
// VOL vsn
// EXT ST,,,CYL,mi
// LBL $SCR1
// LFD DM01
```

Note that the file label is specified as $SCR1, and the logical file description is DM01. The extent statement (EXT) is required to allocate temporary space for the scratch file, and the mi parameter specifies the number of cylinders to be allocated. Finally, the specific amount of work space you need for MILOAD depends on:

- number of key structures involved
- maximum (longest) key length
- total number of records being loaded

Use the following algorithm to determine the amount of work space you need to run MILOAD:

$$S = \frac{(A)\ (B)\ (C)}{D}$$

where:

S
: Is the amount of space needed in terms of 256-byte sectors.

A
: Is the size (maximum length) of the largest key plus 5.

B
: Is the number of key structures.

C
: Is the total number of records being loaded.

D
: Is the physical sector size (256) of the scratch disk.

To convert the space (s) value into cylinders, divide s by the number of blocks per track. Then divide the resultant quotient by the number of tracks per cylinder for your particular scratch disk. Add 20 percent as a size factor.

- Printer File

  You must assign a printer to your job in order to run MILOAD. The device assignment set for the printer is as follows:

```
// DVC 20
// LFD PRNTR
```

7.2.2. Control Statements for Running MILOAD

The control statements for running MILOAD are compatible with those used by the DATA routine for copying MIRAM files. That is, MILOAD uses the input and output utility statement (Uio) and several of its keyword parameters (A, B, FV, L_c, MK_n, MKR_n, and OM) to accomplish the load operation you want performed. These statements define the type of load operation, the input and output device types, and the characteristics of the files being processed.

The Uio statement and its parameters must be included between the start-of-data (/$) and the end-of-data (/*) statements, which follow // EXEC MILOAD in your job stream.

A brief description of the Uio statement and its parameters follows. If you need a more detailed explanation of their function, refer to Section 3.

- Uio Statement

  The Uio statement identifies the device types you require for input and output. It is a mandatory statement that you specify in one of two forms:

  UDD
  : Indicates that MILOAD is to perform a disk input to disk output operation.

  UTD
  : Indicates that MILOAD is to perform a tape input to disk output operation.

  This statement must begin in column 2 or greater and need only be specified once in your job stream. At least one keyword parameter must be included on the same line of code as the Uio statement.

For example:

```
UDD    A=(r,b)
       B=(r,b)
```

Otherwise, MILOAD generates an error.

- A=(r,b) Keyword Parameter

  The A parameter specifies the record size (r) and the buffer size (b) of your input file. This parameter is mandatory if your input file resides on tape. You need not specify this parameter for disk input files. In this case, the record size from the format labels is used; the buffer size is essentially ignored because MILOAD optimizes buffer size to achieve better performance.

  The maximum buffer size for the tape input file is 32,767.

  When your input tape file contains variable-length records, you must specify r as the minimum record length and b as the maximum block length.

- B=(r,b) Keyword Parameter

  The B parameter is optional. When used, it defines the record size (r) and the buffer size (b) for your output file. If omitted, MILOAD uses the record size of your input file as the default value, essentially ignoring the buffer size since it optimizes buffer size for performance. When specifying r, do not include the record control block (RCB) as part of this value.

- FV=(n) Keyword Parameter

  The FV parameter is also optional. When specified, it indicates two things:

  - The record format is variable length.
  - The file slot size is (n)

  Make certain that the slot size (n) you specify is large enough to hold the largest expected input record.

  If omitted, MILOAD assumes a fixed record length format.

- L Keyword Parameter

  The L keyword parameter provides MILOAD with tape input label information. It is mandatory only when your input is on tape. You can specify this parameter in one of two forms:

  `L0`
  Indicates standard tape labeled input.

LC
Indicates unlabeled tape input.

- MKn=(len,loc[,DUP][,CHG]) Keyword Parameter

  The MK parameter is mandatory. It defines the key specifications for your output file. You use it to identify the length and location of each key and to specify whether duplicate keys and key changes are permitted for particular key structures.

  You must include at least one key specification in your MILOAD job stream, but you are limited to a maximum of five specifications (MK1 through MK5) for any one MILOAD run. If you specify more than one key specification, you must define them consecutively in ascending order beginning with MK1, MK2, and so on.

  You can specify a key length (len) from 1 to 80 bytes. The location of that field (loc) is also expressed in bytes. However, the value specified represents the number of bytes preceding the key field in the record, relative to zero. The value you specify for loc must be within the specified record length and must be less than 32,767 (maximum for MIRAM files). Keep in mind that if your input records are of variable length, the 4-byte header must also be considered to determine the actual key location.

  If you want to allow duplicate keys for the particular structure defined, include the DUP subparameter in your specification. Likewise, if you want key changes permitted during updates, include the optional CHG parameter.

- MKR=(n) Keyword Parameter

  The MKR is optional and is used with disk input only. When specified, it allows an existing MIRAM keyed file to be used as INPUT1 and directs MILOAD to read that file in sequence by key of reference n. The range of n is from 1 to 5. This parameter is used to load the OUTPUT1 file in a specific key sequence.

  If omitted, INPUT1 is read as unkeyed.

- OM=(I,n[,V][,R]) Keyword Parameter

  The OM parameter is mandatory. You use it to define the characteristics of your output file; that is, via its subparameters, define the size of the index buffer needed to build your multikeyed output file, define the output as a multivolume file, and specify whether the output file is to contain a record control byte (RCB) for each record.

  You must always include the I subparameter as part of this specification. It identifies your output file as being indexed.

The n subparameter is required to specify the size of the index buffer used to build your multikeyed output file. Buffer size is expressed as the number of 256-byte sectors you need to build your file. You can indicate from 1 to 24 sectors.

The V subparameter indicates that the output file created is a multimount file (all volumes must be on line). Because MILOAD only creates multimount files, V (the default) is always used. Therefore, you can ignore this subparameter.

The R subparameter determines whether the output file contains an RCB. When R is specified, MILOAD creates the output file *without* the RCB included. When R is omitted, the output file is created *with* the RCB included.

The RCB should be included in the output file if the programs that later access this newly created file are concerned with record deletion.

- REBUILD=YES Keyword Parameter

  The REBUILD parameter is optional and is used with disk input only. When specified, it directs MILOAD to rebuild the index on the OUTPUT1 file. OUTPUT1 must be an existing keyed MIRAM file. The data is left intact. All that is necessary to specify for this function is the UDD statement with REBUILD=YES and JCL for OUTPUT1 and DM01. No other keyword parameters are required, and they are ignored if specified. You cannot specify the MKR parameter with the REBUILD parameter. Also, do not specify INIT in the JCL for OUTPUT1 when the REBUILD parameter is specified.

  If omitted, MILOAD runs in its normal mode; that is, loading OUTPUT1 from INPUT1.

7.2.3. Sample Job Stream for Running MILOAD

In the job stream that follows, you will note that the input, output, and scratch files are located on separate devices. This is not a requirement, but is a performance consideration because it reduces unnecessary disk-head movement.

```
// JOB MILOAD,,18000                    Job statement
// DVC 20 // LFD PRNTR                  Printer file
// DVC 61 // VOL DMTST1
// LBL MILOAD.INPUT                     Input file
// LFD INPUT1
// DVC 62 // VOL DMTST2
// LBL MILOAD.OUTPUT                    Output file
// LFD OUTPUT1,, INIT
// DVC 50 // VOL REL071
// EXT ST,,,CYL,30                      Scratch file
// LBL $SCR1
// LFD DM01
// EXEC MILOAD                          MILOAD execution call
/$
```

```
UDD     A=(255,1024)                Input file record and buffer sizes
        B=(255,1024)                Output file record and buffer sizes
        OM=(I,1)                    Output file index buffer size
        MK1=(4,8)                   Primary (first) key specification
        MK2=(13,6,DUP,CHG)
        MK3=(22,10,DUP,CHG)         Additional key specification
/*
/&
// FIN
```

7.2.4. Output Listing Produced by MILOAD

In addition to creating your MIRAM characteristic output file, MILOAD generates an output listing after it successfully terminates. This listing provides you with the following information:

- A statistical summary of the number of records loaded and the number of duplicate keys created for each key structure
- The characteristics of the output file created (including key specifications for all key structures, record size, and buffer size)
- Device and file identification that includes the input device type, output disk type, input and output device volume serial numbers, and input and output file identifiers

Figure 7-1 is a typical example of an output listing produced when MILOAD successfully terminates.

```
JC06    USING    DEV=301  VSN=UMTST1
JC01    JOB MILOAD3  EXECUTING JOB STEP MILOAD00 #002 00:06:19
START   INDEX SORT
END OF INDEX-SORT
01. KEY INSERT STARTED
02. KEY INSERT STARTED
03. KEY INSERT STARTED
 SORT MI00 END OF SORT
SORT A186 RECORDS IN          300 RECORDS DELETED        0
****** NUMBER OF RECORDS LOADED BY MILOAD :        100
AC10  LFD - PRNTR  , FORM NAME - STAND1  , COPIES - 0001, PAGES - 00000001, STEP =002
AC11 STEP #002  (MILOAD00) USED 00064478 BYTES   ELAPSED WALL CLOCK  TIME=00:00:53#336 TOTAL SVC CALLS=00001660
AC12    TERM CODE=000       SWITCH-PRIORITY=10   CPU TIME USED          =00:00:40#671  TRANSIENT CALLS=00000079
AC13 UPSI SETTING X'00'
AC19            DEVICE EXCP's          =00000486       =00000497
AC21 JOB TOTALS    USED 00064478 BYTES      TOTAL ELAPSED WALL CLOCK TIME=00:01:13#128 TOTAL JOB SVC CALLS=00002331
AC22                                        WALL CLOCK TIME OF ALL STEPS =00:01:06#319 JOB TRANSIENT CALLS=00000131
AC23                                        TOTAL CPU TIME OF ALL STEPS  =00:00:47#227 TOTAL JOB EXCP'S   =00001232
JC02 JOB MILOAD  TERMINATED NORMALLY                hh:mm:ss
A=(255,1024),B=(255,1024)
OM=(I,1)
MK1=(4,8,DUP,CHG)
MK2=(13,6,DUP,CHG)
MK3=(22,10,DUP,CHG)
```

Figure 7-1. Typical Output Listing for MILOAD (cont.)

```
                  *****    MIRAM LOADER PROTOCOL    *****
INPUT DEVICE           =  DISK
OUTPUT DISC-TYPE       =  8418
INPUT FILE-NAME        =  MILOAD.INPUT
INPUT VOLUME-NAME      =  DMTST1
OUTPUT FILE-NAME       =  MILOAD.OUTPUT
OUTPUT VOLUME-NAME     =  DMTST2
INPUT RECSIZE          =     255
OUTPUT RECSIZE         =     255
RECORD CONTROL BYTE    =     YES
KEY NUMBER 1 LENGTH    =       4     DUP=Y     CHG=Y
KEY NUMBER 1 LOC       =       8
KEY NUMBER 2 LENGTH    =      13     DUP=Y     CHG=Y
KEY NUMBER 2 LOC.      =       6
KEY NUMBER 3 LENGTH    =      22     DUP=Y     CHG=Y
KEY NUMBER 3 LOC.      =      10
INDEX SIZE             =     256
COUNT OF DUPLIC KEYS:    KEY1    KEY 2    KEY 3
                           0        0        0
NUMBER OF RECORDS LOADED :          100
DATE :  yy/mm/dd    TIME :  hh:mm:ss
```

Figure 7-1. Typical Output Listing for MILOAD

7.2.5. Examples of Typical MILOAD Jobs

The following examples give five variations of using MILOAD for loading MIRAM files.

Example 1. Disk Input to Disk Output (Multidrive)

In this example, MILOAD creates a 2-key output file with an RCB. Duplicate key and key changes are not allowed. The file contains fixed-format records. The record size is 256, and the input and output files are on separate volumes.

```
// JOB MILOAD1,,18000                          Job statement
// DVC 20    // LFD PRNTR                      Print file
// DVC 50    // VOL VOL742                     } Input file
// LBL INFILE    // LFD INPUT1                 }
// DVC 51    // VOL VOL692                     } Output file
// LBL OUTFILE   // LFD OUTPUT1,,INIT          }
// DVC 50    // VOL VOL742                     }
// EXT ST,,,CYL,50                            } Scratch file - temporary work space
// LBL $SCR1    // LFD DM01                   }
// EXEC MILOAD
/$
    UDD    A=(255,256),B=(255,256)             }
           OM=(I,1)                            } Utility input and output statement
           MK1=(8,4)                           }
           MK2=(6,13)                          }
/*
/&
// FIN
```

Example 2. Disk Input to Disk Output (Single Drive)

In this example, MILOAD creates a 2-key file with no RCB. Duplicate keys and key changes are allowed on Key 2 only. The MIRAM file produced contains variable-size records, and the slot size for these records is 100 bytes.

```
// JOB MILOAD2,,18000                          Job statement
// DVC 20         // LFD PRNTR                 Print file
// DVC 50         // VOL VOL742                Input file
// LBL INFILE     // LFD INPUT1
// DVC 50         // VOL VOL742                Output file
// LBL OUTFILE    // LFD OUTPUT1,,INIT
// DVC 50         // VOL VOL742
// EXT ST,,,CYL,35                            Scratch file - temporary work space
// LBL $SCR1      // LFD DM01
// EXEC MILOAD
/$
```

```
    UDD     A=(100,512)
            B=(100,512)
            OM=(I,1,V,R)                    Utility input and output
            FV=(100)                        statement
            MK1=(8,4)
            MK2=(16,8,DUP,CHG)
/*
/&
// FIN
```

Example 3. Disk Input to Disk Output (MKR Parameter)

In this example, MILOAD will read INPUT1, which is an existing keyed MIRAM file, and generate OUTPUT1 in Key 3 sequence.

```
// JOB MKRUSE                                   Job statament
// DVC 20            // LFD PRNTR                Print file
// DVC 50           // VOL PACK01
// LBL KEYED.FILE   // LFD INPUT1               Input file
// DVC 51           // VOL PACK02
// LBL NEW.KEYED.FILE  // LFD OUTPUT1,,INIT     Output file
// DVC 50           // VOL PACK01
// EXT ST,,,CYL,10                             Scratch file - temporary work space
// LBL $SCRI        // LFD DM01
// EXEC MILOAD
/$
 UDD  A=(255,256)
      B=(255,256)
      OM=(I,1)
      MKR=(3)                                   MILOAD parameters including MKR
      MK1=(8,4,DUP)
      MK2=(6,13,DUP,CHG)
      MK3=(10,20)
/*
/&
// FIN
```

Example 4. Rebuild Existing Miram Indexed File

In this example, MILOAD will rebuild a MIRAM file in place.

```
// JOB REBUILD                                    Job statement
// DVC 20           // LFD PRNTR
// DVC 50           // VOL PACK01                  Input/Output file
// LBL KEYED.FILE   // LFD OUTPUT1
// DVC 50           // VOL PACK01
// EXT ST,,,CYL,5                                 Scratch file - temporary work space
// LBL $SCR1        // LFD DM01
// EXEC MILOAD
/$
 UDD REBUILD=YES                                  Single MILOAD parameter
/&
// FIN
```

Example 5. Tape Input to Disk Output

In this example, MILOAD creates a 5-key MIRAM output file with a record size of 50. No RCB, fixed format, duplicate keys are allowed on Keys 2 through 5, and key changes are not allowed on any key. The tape input is standard labelled.

```
// JOB MILOAD3,,18000                             Job statement
// DVC 20          // LFD PRNTR                   Print file
// DVC 90          // VOL TAP001                  Input tape file
// LBL TAPFILE     // LFD INPUT1
// DVC 50          // VOL VOL742                  Output disk file
// LBL OUTFILE     // LFD OUTPUT1,,INIT
// DVC 50          // VOL VOL742
// EXT ST,,,CYL,70                                Scratch file - temporary work space
// LBL $SCR1     // LFD DM01
// EXEC MILOAD
/$
   UTD    A=(50,50)
          B=(50,512)
          OM=(I,1,,R)
          L0
          MK1=(6,14)                             Utility input and output
          MK2=(10,4,DUP)                         statement
          MK3=(17,6,DUP)
          MK4=(4,12,DUP)
         MK5=(5,27,DUP)
/*
/&
// FIN
```

7.3. Messages - Interface for Users and Operators

MILOAD provides both informational and error messages to assist you in using and understanding it. All messages are displayed on your console screen and recorded on your printer listings.

7.3.1. Informational Messages

Informational messages inform you of the current status of MILOAD during execution. They indicate when an important process begins. For example, MILOAD uses the SORT routine to sort the scratch file containing all the file keys. When this process begins, a SORT called informational message is displayed. An informational message is also displayed indicating that MILOAD has completed the sorting and is beginning construction of an indexed key structure. No action is required after these messages. They are strictly for your information, so you know the status of MILOAD and when it completes.

7.3.2. Error Messages

If your job does not run successfully and is terminated by MILOAD, one or more error messages are generated and displayed explaining why the run was unsuccessful. The messages are self-explanatory and can be used to remedy the error condition. The error messages are listed directly below the run statements on the output listing generated by MILOAD and are formatted as follows:

```
DUFnn - { W } - message text
        { S }
```

where:

nn
: Is the message number.

W
: Indicates that the message is a user warning.

S
: Indicates that the error is serious and that the UPSI byte is set to X'40'.

For a complete listing of all MILOAD error messages, refer to the *System Messages Operations Reference Handbook* (UP-8076).

7.3.3. Unrecoverable Error Conditions

In addition to the error messages, there are five possible CANCEL exit paths that signify unrecoverable errors. They are identified as follows:

Open error	Results in a DUF45 error message and CANCEL termination
Error writing index	Results in a DUF28 error message and CANCEL termination
Error reading INPUT1	Results in a DUF29 error message and CANCEL termination
Error writing OUTPUT1	Results in a DUF30 error message and CANCEL termination
Error in sort	Results in a DUF45 error message and CANCEL termination

7.4. Additional Features of the MILOAD Utility

In addition to loading MIRAM files, MILOAD can bulk extend (add large numbers of records to) an existing MIRAM file. The same restrictions and constraints previously noted for file creation apply to this added capability of MILOAD. Performance varies for bulk extending and is dependent upon the relationship between the key values of the records being added to those of the records already existing in the file.

Before attempting to bulk extend an existing file, we recommend that you do two things. First, make certain that the records being added to the existing file are not in conflict with that file. That is, make sure that you have not defined file specifications to MILOAD that differ from those which exist for the existing MIRAM file. An example of these differences could be, format differences, RCSZ differences, key differences, and so forth. Second, make a backup copy of the existing file. This will allow you to restore the original contents of the file, should MILOAD terminate your job due to an error condition. Use the DUMP/RESTORE facility to back up your existing file. Output file shareability during bulk extending is not permitted. MILOAD requires the file exclusively until it has successfully completed.

Appendix
Data Utility Input and Output Parameters for Mixed Systems

Users of System 80 models 7E through 20 have the option of running in a "mixed" data management mode, that is, using both consolidated data management and basic data management. Series 90 users migrating to System 80 models 7E through 20 can use both the MIRAM access method and those access methods supported by basic data management (SAM, DAM, ISAM, IRAM).

Table A-1 lists the basic data management keyword parameters needed by these users for SAM, DAM, ISAM, and IRAM files.

Table A-1. Additional Keyword Parameters for Data Utility Input and Output Statements for Mixed Systems

Keyword Parameters	Description
A=(r,b)	INPUT1 Block and Record Length For fixed-length (FF) records on 8413 card or tape files, this parameter specifies record length (r) and block length (b) expressed in decimal. If omitted, r and b both default to 80. The block length b must be a multiple of r. For variable-length (FV) records on tape input file, this parameter specifies a minimum r and a maximum b. If omitted, the minimum r defaults to 0 and the maximum b defaults to 80. For fixed-length (FF) records on SAM, DAM, or ISAM disk input files, this parameter is not used. The r and b lengths are taken from the VTOC entry for the INPUT1 file. If this parameter is specified for SAM, DAM, or ISAM files, no error message will be generated but the information specified in this parameter will be ignored. If DAM input is specified in the preceding case, the r represents the record data length, the key length is not included in r. If SAM input is specified, then b must be a multiple of r. continued

Table A-1. Additional Keyword Parameters for Data Utility Input and Output Statements for Mixed Systems (cont.)

Keyword Parameters	Description
	For fixed-length (FF) records on IRAM or MIRAM files, this parameter specifies a dummy value less than six digits long (r) and the buffer length (b) expressed in decimal (b must be a multiple of sector size). If the specified buffer length is less than the minimum allowed for the record size, as specified in the VTOC entry for INPUT1, then the minimum will be used. For variable-length (FV) records on SAM, DAM, or ISAM files, this parameter specifies a minimum r and a dummy b value less than six digits long. The lock length will be taken from the VTOC entry for INPUT1. For variable-length (FV) records in IRAM or MIRAM files, this parameter specifies a minimum r and the buffer length (b must be a multiple of sector size). If the specified b is less than the maximum allowed for the record size, as specified in the VTOC entry for INPUT1, then the minimum is used. All subparameters to this parameter have a maximum size limit of 5 decimal digits.
B=(r,b)	OUTPUT1/INPUT2 Block and Record Length For fixed-length (FF) records on 8414 card tape and SAM disk files, this parameter specifies record length (r) and block length (b). For these files, b should be an even multiple of r. For fixed-length (FF) records on DAM disk files, this parameter specifies r, which excludes key length, and b, which is data length plus key length. For fixed-length (FF) records on ISAM disk files, this parameter specifies r, which includes key length, and b, which includes ISAM block overhead of two bytes per block plus five bytes per record. For fixed-length (FF) records on IRAM/MIRAM disk files, this parameter specifies r, which includes key lengths but excludes MIRAM record control byte, and buffer length (b), which must be a multiple of sector size (sector size is normally 256 bytes). If omitted for fixed-length (FF) records, r defaults to the INPUT1 file record length (from the A=( ) parameter for nondisk input or the file format labels for a disk input file). Block/buffer length (b) defaults to the INPUT1 file block/buffer length (from the A=( ) parameter for nondisk and IRAM/MIRAM input files or the file format labels for all other disk input files).

continued

Table A-1. Additional Keyword Parameters for Data Utility Input and Output Statements for Mixed Systems (cont.)

Keyword Parameters	Description
	For variable-length (FV) records on tape and SAM disk files, this parameter specifies minimum record length (r), which includes a 4-byte variable record descriptor word, and block length (b), which must be at least the maximum record plus four bytes for the variable block descriptor word. For variable-length (FV) records on DAM disk files, this parameter specifies minimum r, which includes the 4-byte variable record descriptor word but excludes key length and b, which is maximum record length plus key length plus four bytes for the variable block descriptor word. For variable-length (FV) records on ISAM disk files, this parameter specifies minimum r, which includes key length plus two bytes for the ISAM variable record descriptor word and b, which is at least maximum record size plus seven bytes for ISAM block overhead. For variable-length (FV) records on MIRAM disk files, this parameter specifies minimum r, which includes all keys plus the 4-byte variable record descriptor word, and b, which must be a multiple of sector size (sector size is normally 256 bytes). If omitted for variable-length (FV) records, the minimum r defaults to zero and the block/buffer length b defaults to the INPUT1 file block/buffer length. If comparing (K2) files and the INPUT2 file is a disk file, the record length for fixed record (FF) files and the block length for fixed (FF) and variable (FV) record files will be obtained from the disk file format labels and override the information in this parameter. For IRAM and MIRAM files, buffer size may be specified here. If it isn't, the minimum allowed is used. This is a performance consideration.
B=(r,p)	For fixed-length (FF) records on the primary printer file (UiP), this parameter specifies the record length (r), and the print line size (p). If this parameter is omitted in this case, r will default to the INPUT1 record length, and a 120-character print line size (p) will be assumed. If r is greater than the record length of the INPUT1 file, the OUTPUT1 record will be padded on the right with blanks. If the specified r is less than the record length of the INPUT1 file, the INPUT1 record length will be used. If first character forms control (SA) is specified, print line size (p) must include one byte for the control character.

continued

Table A-1. Additional Keyword Parameters for Data Utility Input and Output Statements for Mixed Systems (cont.)

Keyword Parameters	Description
	For variable-length (FV) records on the primary printer file (UiP), this parameter specifies a minimum and a print line size (p). The minimum r cannot be less than 2. If this parameter is omitted in this case, the minimum r will default to the value used for the INPUT1 file, and a 120-character print line size will be assumed.
G=(nn)	ISAM Percent Cylinder Overflow This parameter specifies the percentage of a cylinder (nn) reserved for cylinder overflow for ISAM output files. The maximum percentage of a cylinder that can be reserved for overflow is 80 percent. If this parameter is omitted, 10 percent is assumed. If zero percent cylinder overflow is used to create the file, then the file can never be updated (i.e., new records can never be inserted in the file).
	Output Disk File Type
OI	Indicates an ISAM OUTPUT1/INPUT2 disk file. If copy (K1) is specified, then the file is created without write verify active.
OIY	Indicates an ISAM OUTPUT1/INPUT2 disk file. If copy (K1) is specified, then the file is created with write verify active.
OS	Indicates a SAM OUTPUT1/INPUT2 disk file. If file copy (K1) is specified, the OUTPUT1 is created without write verify.
OSY	Indicates a SAM OUTPUT1/INPUT2 disk file. If file copy (K1) is specified, the OUTPUT1 is created with write verify.
OR=(C[,V])	Indicates an IRAM OUTPUT1/INPUT2 disk file is to be created without write verify active. When this parameter is specified, the first subparameter must also be specified; all other subparameters are optional. These may or may not have commas present to indicate omitted parameters. The subparameters for OR=( ) are as follows: C Indicates a consecutive IRAM file and the file has no INDEX partition. If C is specified, the "p" and "S" entries must not be specified in the OR=( ) parameter.

continued

Table A-1. Additional Keyword Parameters for Data Utility Input and Output Statements for Mixed Systems (cont.)

Keyword Parameters	Description
I	Indicates the output file is to be created with an index partition.
p	Specifies the index block size for this file. It is a 1-digit decimal number from 1 through 7 representing the number of 256-byte multiples in the index block. The index block size used here stands for the life of the file. The default value is 1.
S	Indicates a sequence check on key values is performed during the creation of this file. If not specified, no sequence check will be performed.
V	Provides IRAM/MIRAM file generation as either single or multivolume mount. If specified, the volume mount indicate is set so that data management requires all volumes of the file to be online at the same time. Although a multivolume mount file is the only type that can be processed randomly, the initial file allocation can never be extended. On the other hand, a single-volume mount file cannot be processed randomly, but can be extended. This also applies to a file occupying only one volume. If this parameter is omitted and input is disk, the default is the input file single/multivolume mount indicator. If input is card, tape, or diskette, single-volume mount output is assumed.
OD	Indicates a DAM OUTPUT1/INPUT 2 disk file is to be created. If copy (K1) is specified, the OUTPUT1 file will be created without write verify.
ODY	Indicates a DAM OUTPUT1/INPUT2 disk file is to be created. If copy (K1) is specified, OUTPUT1 file will be create with write verify.
ORY=(C[,V]) or ORY=(I[,p][,S][,V])	Indicates an IRAM OUTPUT1/INPUT2 disk file is to be created. If copy (K1) is specified, the OUTPUT1 file will be created with write verify active. C, I, p, S, and V have the same function as they have in the OR=( ) parameter.
	Key Length
V=(nnnnn)	For a file copy (K1), this specifies the key length (less than 6 decimal digits) for the OUTPUT1 (UiD) disk file. If this parameter is omitted and the INPUT1 file is a disk file (UDD and K1 are specified), the INPUT1 key length specified in the INPUT1 VTOC entry is used. Otherwise, a 10-byte key is assumed. If file compare (K2) is specified, the key length is taken from the VTOC entry for the INPUT2 file.

continued

Table A-1. Additional Keyword Parameters for Data Utility Input and Output Statements for Mixed Systems (cont.)

Keyword Parameters	Description
	Since the key lengths for all input disk files are taken from the VTOC entries for the corresponding files, no input disk key length parameter is provided. Data management supports keys only for disk files.
V=(mmmmm,nnnnn)	This form of the parameter also is accepted. It has the same function as the V=(nnnnn) parameter except that mmmmm is a dummy entry (less than 6 decimal digits) that is always ignored. The second entry, nnnnn, has the same function and defaults as in the V=(nnnnn) parameter.
	Key Location (DAM/ISAM/IRAM files)
W=(nnnnn)	For a copy operation (K1), this specifies the key location (the number of bytes that precede the key) for the OUTPUT1 file (UiD). If this parameter is omitted and the INPUT1 file is a disk file (UDD and K1 are specified), the INPUT1 key location as specified in the INPUT1 VTOC entry is used. Otherwise, a key location of 0 is used for fixed-length (FF) files, a key location of 2 is used for ISAM variable (OI and FV specified) files, and a key location of 4 is used for all other variable-length disk files. For file compare (K2), the key location is taken from the VTOC entry for the INPUT2 file.
W=(mmmmm,nnnnn)	This form of the parameter also is accepted to indicate the key location. It has the same function as the W=(nnnnn) parameter, where mmmmm is a dummy entry that is always ignored. The second entry, nnnnn, has the same function and defaults as in the W=(nnnnn) parameter.

Index

continued

E

F

H

I

J

K

L

M

N

O

T

U

V

W

Z

UNISYS

USER COMMENTS

We will use your comments to improve subsequent editions.

NOTE: Please do not use this form as an order blank.

_____ _____

(Document Title)

_____ _____ _____

(Document No.) *(Revision No.)* *(Update Level)*

Comments:

From:

_____

(Name of User)

_____

(Business Address)

Fold on dotted lines, and mail. (No postage is necessary if mailed in the U.S.A.)
Thank you for your cooperation

FOLD

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

Unisys Corporation
E/MSG Product Information Development
PO Box 500 — E5-114
Blue Bell, PA 19422-9990

FOLD

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

First Class Permit No. 21 Blue Bell, PA

Postage Will Be Paid By Addressee

Unisys Corporation
OS/3 Systems Product Information Development
PO Box 500 - E5-114
Blue Bell, PA 19422-9990